EEEM066
Deep Convolutional Neural Networks for Fine-Grained Knife Classification

A disciplined approach to inductive transfer learning

Elena Mylona, URN: xxx, EM02069@surrey.ac.uk

Abstract

The global increase in knife-related crimes, especially in England and Wales where offences rose by 9% in the year ending March 2022, underscores the necessity for a deep learning weapon analysis. Thus, we propose our custom-designed KnifeNet model based on the DeiT-3 transformer tackling this weapon inaccuracy issues for knives. In this study, the composed parts of inductive transfer learning, pre-training and fine-tuning hyperparameters of pretrained neural networks and transformers, have been applied to the above formulated problem of knives achieving optimal accuracy for our multi-class knife classification system. Ablation studies were conducted for fine grained knife recognition powered by deep learning. In our experiments we show the results that EfficientNet, DenseNet and DeiT pre-trained models after fine-tuning. We scrutinised our experimental results and analysed the performance metrics obtained from the final testing and compared altogether. Our results recorded peak test mAP with the best deep ConvNet model, DenseNet264 with 0.6785 and our KnifeDeiT-3 reaching 0.7316 mAP values.

1. Introduction

Convolutional neural networks (CNNs) have represented the foremost approach in machine learning for visual object recognition [1]. In recent years, deep learning models were deployed to tackle classification problems with the use of CNNs being involved in the existing approaches. To develop our classification system, we utilised deep CNNs which were prior trained on the ImageNet dataset, namely EfficientNet-B0, EfficientNet-B6 as well as DenseNet-264 and the DeiT-3 transformer. In our approach, we adopt the common procedure of inductive transfer learning, emphasising on fine-tuning the hyperparameters of neural networks to achieve optimal accuracy of the model which well suits our formulated problem of a multi-class knife image classification system.

We discarded the last fully connected layers for classification of the different pre-trained models that are being used for these experiments. Added new layers for the specified 192 classes of knives, and then re-trained those new convolutional layers along fine-tuned their hyper-parameters. The training dataset consists of around 9928 knife images in 192 classes. The test dataset consists of 351 images. The performance of the models is evaluated using the mean average precision (mAP) for each of our numerous experiments.

The caveat is that the training procedure for complex networks such as the model variant DenseNet264, from the Densely Connected Convolutional Networks family, with its 264 layers, dense blocks for maximum information flow between layers and its growth rate, required up to 4 hours on Nvidia's T4 GPU in Google Collaboratory notebook to take place.

1.1. Deep CNNs Model Architectures

In neural networks (NN), weights and biases guide data flow during forward propagation. Afterward, backward propagation refines connections using errors from the forward phase, adjusting nodes and connections to improve accuracy while networks make accurate predictions [2].

Training strategies have been applied to all previously mentioned models with the purpose of cultivating highly discriminative CNNs for representation learning.

The primary challenge in training a deep ConvNets was linked to the scheduling of the learning rate and the degree of L2 weight decay regularisation invoked.

We experimented with freezing some neural network layers to stabilise learning. Additionally, we adjusted dropout rates in certain layers, ranging from 0.3 to 0.7, to prevent overfitting. Depthwise convolutional 2D layers were also added strategically to enhance the model's performance.

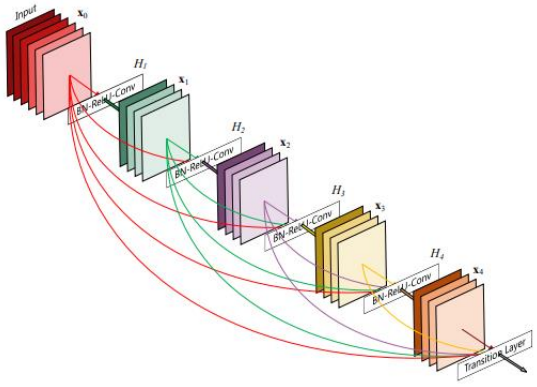


Figure 1: A 5-layer dense block with a growth rate of k = 4.

Contemporary research suggests that the Densely Connected Convolutional Networks (DenseNets) stand out due to their unique architecture of connectivity patterns [7]. As illustrated schematically in Figure 1, each layer of a dense block receives all preceding feature maps as its input and then passes them to all subsequent layers in order to preserve its feed-forward nature. This structure has impressed us and prompted the revelation of its results on our formulated problem of knives. The primary advantages of DenseNets include strengthening the propagation of features, alleviating the problem of vanishing gradients, and substantially reducing the number of parameters, leading to more improved performance [7].

It has been shown that Vision Transformers (ViT) can outperform existing CNNs [9]. Our entirely modified network uses as baseline the Data efficient image Transformer III, known as DeiT-3, which was pretrained on ImageNet-22k and fine-tuned on ImageNet-1k with 86.6 million of Params and 23.9 million Activation values. Hence, this large number of neurons spread across the various layers of the network convinced us to compare it with the recent deep architectures used in our investigation.

1.2. Dataset Segregation

Initially, the whole dataset was partitioned into three distinct subsets, comprising a training and a validation set, with the inclusion of a separate testing set. In particular, the proportion of the test dataset constitutes 351 knife images, which accounts for a mere 3.41% of the aggregate dataset when combined with the 9928 images in the training set. Concomitantly, the validation set comprises 401 images forming approximately 4.04% of the total training data.

A key characteristic of the test dataset that we strictly adhered to is being separate and distinct from the training dataset, which aids in measuring deep ConvNets' ability to generalise effectively to novel, unobserved knife instances. A simple testing pipeline for knife images was executed without applying any transformations on them except converting them to the appropriate floating-point data type for the standard input of image tensors into our modified deep neural networks. In detail, distinct pipelines for training, validation and testing were implemented using the relevant CSV files that were subsequently supplied to their respective data loaders to acquire accuracy and mAP value.

Prepared the validation and test data by preprocessing and not augmenting our knife images. It involved maintaining the images to a consistent size, adhering to the same dimensions, 224×224, converting them to tensors, and normalizing their pixel values solely for validation as it aligns the input data distribution with the distribution of the data images used during the pretraining of the models on ImageNet dataset. Hence, the input data remains in the correct format and scale for the models but do not introduce the variability or new samples characteristic of data augmentation since it does not randomly alter or expand the dataset of knife images.

1.2.1 Samples of Images with Noise

Training models with a diverse dataset like KnifeNet-10k, indeed posed multiple challenges, as one of them is illustrated in Figure 2. These types of images are well-suited for deep convolutional neural networks and vision transformers for classification task given their background distractions, distracting texture variance and variations in illumination.

Textural noise from human figures and hands, pallet cases, sheaths, tree nodes, red asphalt paint lines, black pavement bricks, to vegetables and meat can be discovered.

Figure 2: Knife Instances of the Training KnifeNet-10k set.

These were detected within the KnifeNet-10k dataset, that can serve as significant distractors for a CNN and ViT to correctly identify the target in an image. Therefore, we attempted to minimise the impact of them by employing data augmentation approaches in a way that irrelevant objects and the busy backgrounds of the images will not detract the networks and transformer from the knife itself.

1.3. Hyperparameters for Fine Tuning

Regarding the related work of our transfer learning approach, we delineate the systematic adjustments made to parameters which handle the training process of the networks on the KnifeNet-10k dataset. Moreover, our experimentations involved fine-tuning the following hyperparameters of neural networks and transformers as well as discussing their impact on training the various ConvNets fine-grained knife classification models.

Notably, a challenging and time-consuming endeavor was to estimate optimal values for the learning rate with the aim of reaching convergence to a global minimum. It was discerned that adjusting the batch size affects the consistency of the learning process, with larger batches often leading to more stable but potentially slower convergence.

Based on the performance metrics obtained in Section 5 from the validation set, we iteratively refined the models until they attained performance levels deemed satisfactory.

This iterative process continues until the model attains a performance level deemed satisfactory. In essence, multiple attempts on hyperparameters configurations have been applied to the models to learn the very fine-grained differences among knife classes.

Initially, we outline the notations and the empirical settings, subsequently elucidating our findings concerning momentum, the practical learning rate, and regularisation techniques including $L1$, $L2$ and dropout. The Nesterov's Accelerated Gradient optimisation method is described as follows [3]:

$$v_{t+1} = \mathrm{m}v_t - \frac{\eta_t}{n}\sum_{i=1}^{n}\nabla\big(\ell(f(x_i, \theta_t + mv_t), y_i)\big)$$

$$\theta_{t+1} = \theta_t + v_{t+1} - \eta\lambda\theta_t$$

"where θ_t indicates the model parameters at iteration t. The hyperparameters include the learning rate η_t, batch size n, momentum coefficient $m \in [0, 1)$, and the weight decay λ".

In our optimisation practices, we attempted to speed up the convergence of this gradient descent by incorporating weight decay within Adam, AdamW optimisers and momentum solely into SGD for preventing overfitting and accelerating the optimisation algorithms.

Our goal is to minimise the loss function ℓ while training deep models and concurrently ensuring a smoother convergence trajectory by building upon the accumulated gradients from our previous steps.

The learning rate, $\boldsymbol{\eta_t}$, an essential hyperparameter that we considered, determines the step size during the gradient descent optimisation process. Therefore, we can control how much the weights in the neural network by are updated during training. Within our experimental framework, we established learning rates to fall within a specific range as shown below:

Learning Rate
0.00005
0.0001
0.0002
0.0005
0.001
0.002
0.005
0.01

Table 2: Learning Rate Hyperparameters for Fine-Grained Knife Classification

The following are the key findings listed on the optimal learning rates for efficient model training:

Small Learning Rate: Progress is slow, and it takes a long time to converge to the minimum of the loss function.

Large Learning Rate: The updates may be too large, causing the optimisation to overshoot the minimum, leading to possible divergence.

Optimal Learning Rate: Achieves efficient convergence to the minimum, balancing the size of steps to neither be too small nor too large.

If the learning rate is changed, we need to adjust:

Batch Size: A smaller learning rate might benefit from a larger batch size and vice versa.

Number of Epochs: With a smaller learning rate, we might need more epochs to converge.

Learning Rate Schedule: Implementing a schedule that adapts the learning rate during training can help with convergence.

An example set of training refinements is illustrated below:

Learning Rate	Batch Size	Epochs	Dropout	Momentum	Weight Decay
0.001	32	10	0.5	0.9	0.0001
0.01	64	20	0.4	0.8	0.0005
0.0001	128	30	0.3	0.7	0.00005
0.0005	256	40	0.2	0.6	0.00001
0.002	512	50	0.1	0.5	0.0002
0.005	1024	60	0.7	0.85	0.0003

Figure 3: Hyperparameters and Configuration Settings.

Must be noted here that momentum is solely integrated into SGD optimiser to improve the convergence of stochastic gradient descent. Investigation concluded that these hyperparameters are interdependent, and changes to one can necessitate adjustments to others while taking into account configurations of the training procedure.

1.3.1 Cosine Annealing Scheduler

In regard to constructing our deep ConvNet models for accurately classifying knives, we employed a Cosine Annealing Scheduler for effective learning rate adjustment. This scheduler modulates the learning rate following a cosine curve, gradually reducing it from a maximum to a minimum value. This approach aids the models in escaping local minima and facilitates convergence to a more optimal solution.

1.3.2 Weight Decay

It is widely known as L2 regularisation, where the weight decay, $\boldsymbol{\lambda}$, encouraged our models to learn more generalizable representations of knife images. In this way, we penalised the models for having large weights, thereby steering the towards simpler, more universal features in the images [5]. The specific values used in our experiments for decay are detailed in the above table of Figure 3. This table provides a clear overview of how we adjusted the weight decay parameter across different configuration settings in our study.

1.3.3 Momentum

Enhancement over stochastic gradient descent (SGD) is essential to consider for our training experiments as momentum, $\boldsymbol{m}$, helped accelerate SGD in the relevant direction and dampens oscillations. Prior conducting

further experiments on the deep models, we proceed with the use of momentum with the aim of escaping any upcoming issues of local minima. The latter are points in the loss landscape where the algorithm converges prematurely, leading to suboptimal solutions.

The moment coefficient is slightly altered by setting it within the interval of [0.5, 0.9] values. We incorporated it in our experiments as shown in Figure 2 to escape local minima and indeed it influenced the learning process when EfficientNet-B6 and DenseNet264 have been being trained. In particular, we recorded convergence on DenseNet264 when values as low as 0.3 to 0.7 were employed for momentum.

1.4. Adaptive Moment Estimation

In the quest to minimise losses in deep learning, three main optimizers such as Adam and AdamW (Adam with Weight Decay) are pivotal, yet our experiment suggest that the SGD optimizer led to better generalisation. This observation intimates that L2 regularisation does not perform optimally within Adam compared to SGD, potentially making stochastic gradient descent a more effective choice for models requiring robust applicability to unseen knife images.

The following graph presents the comparison between two optimisation algorithms, Adam and AdamW, plotting their training loss on cross-entropy against test error rates. The trend shows that as training loss decreases, the test error also reduces, with AdamW consistently performing slightly better than Adam, as evidenced by the lower error rates for the same training loss values.

2. Ablation Studies

This ablation study dissects the individual and collective influence of hyperparameters, optimisation algorithms selection, deep ConvNets and ViT architectures along with their configuration settings, and ensemble methods on the overall efficacy of the pretrained models. In every DL-based classification model, two internal components are present: the feature extraction portion, stacking of layers of convolution and pooling, and the classification segment of fully connected (FC) layers [3]. The application of transfer learning when constructing our knife classification model, we identified an uncertainty regarding which of these parts is more critical for effective adaptation to a new dataset.

For our formulated fine-grained object classification problem, we monitored the fluctuations of losses for training and validation at each iteration to comprehend the models' behaviour and whether it leads to overfitting or stagnation. To effectively train deep models such as EfficientNet-B0 and B6, DenseNet264, and a novel designated network using DeiT-3 as a baseline model, namely OurKnifeNet in this paper, we undertook extensive modifications to the models' architecture and fine-tuning of hyperparameters, guided by the validation mean average precision values. OurKnifeNet, underwent significant alteration, with more than half of its neurons removed by defining a dropout rate 0.7, as it is already a complex network to train for our focused task of knife classification.

During these processes, batch normalisation was strategically applied to the network layers to stabilise learning and improve convergence. Depthwise convolutional layers were integrated to enhance the model's efficiency without compromising accuracy. The Squeeze and Excitation Network (SENet) block was also introduced, contributing a recalibration mechanism for feature channel refinement "by explicitly modelling interdependencies between channels", enhancing the network's representational capacity [4].

Furthermore, the internal parameters, including the weights of each neural network architecture, were meticulously optimized to enhance the model's ability to classify knife instances accurately within the training dataset. In addition to these architectural tweaks, we implemented backpropagation techniques for learning optimisation and removed certain class weightings from the cross-entropy loss function to mitigate overfitting. We experimented with different model architectures, EfficientNet-B0, B6, DenseNet121, DenseNet264, and a custom-designed network based on DeiT-3, testing variations in the number of neural layers and their configurations to gauge their performance impacts.

The nature of all these procedures was inherently iterative, encompassing multiple cycles of training, validation and subsequent adjustments. This iterative refinement continued until each model reached a performance threshold of mean average precision, mAP, that aligns with the criteria of adequacy. The experimental results from our comparative model performance metrics, robustly corroborate the efficacy of DenseNet264, demonstrating its suitability in fine-grained knife classification powered by deep ConvNets. Subsequently, we did not achieve compelling results during our exploration phase on EfficientNet-B0 and DenseNet121 variants, yet both DenseNet264 and OurKnifeNet outperformed the rest of the experimented deep models of our ablation studies.

2.1. Loss Function Alignments

A decision was made to use Cross Entropy Loss as the primary loss function for our specialised task of knife multi-class classification. This standard loss function is vital in evaluating our knife classification models, particularly those outputting probabilistic predictions between 0 and 1. Cross Entropy loss quantifies the

deviation of predicted probabilities from their corresponding actual labels, with the loss escalating as this divergence increases.

In our implementation, Cross Entropy Loss was a natural choice because it combines both negative log-likelihood loss and softmax functionality. It adeptly handles multiple classes by contrasting predicted knife class probabilities against actual knife class labels, averaging the loss over all instances in the test set. This makes it particularly suitable for the demands of our knife classification project.

Therefore, we experimented with the tensor of weights passed to the Cross Entropy Loss as the weight argument, and without any class weighting. For the first, the weights adjusted the loss based on the knife class, which we used to handle class imbalance where some knife classes were underrepresented in the training data. Consequently, this type of weighted loss is then scaled up, making our fine-tuned deep ConvNets and ViT models pay more attention to these sensitive knife classes.

2.2. The Significance of Random Seed Setting

This seed enabled us to iteratively refine our pretrained models with the assurance that improvements were due to our modifications rather than variations of randomness. We aimed to make our evaluation outcomes deterministic and reproducible with a seed value being set to 42, attributing improvements solely to our model refinements, and not to random variations.

Without setting a seed value, our entire framework hinges on stochastic variability processes such as weight initialisation and data shuffling. Consequently, the mAP values exhibited inconsistency after training we observed that the mAP values were not consistent during training and validating when no configurations or modifications took place. This stems from the random processes which are driven by pseudo-random number generators, and they are fundamental in navigating the vast solution space of NN parameters.

Regarding our updated now reproducible results, it is led to marked performance gains in DenseNet121, DenseNet264 and OurKnifeNet with mAP improvements of 11.2%, 3.3%, 1.9%, respectively.

2.3. Data Augmentation

In the preprocessing stage, we applied random transformations to training data to generate new and varied samples, which helps in preventing potential overfitting and improving models' generalisation capabilities. Particularly, the images undergo normalisation using the mean and standard deviation values derived from the ImageNet dataset. Our data augmentation strategy aligns with established methods typically employed for training models on ImageNet, specifically random mirroring, random scaled cropping incorporating variations in scale and aspect ratio, and color jittering. Subsequently, the augmented images are resized to a resolution of 224×224 pixels.

Since some classes consist of fewer knife images than other classes, we conclude that we must make the models less sensitive to small changes within the same knife class. We turned our focus towards intra-class variance to combat class imbalance of knives through advanced data augmentations methods and inter-class discrepancy [8] by modifying the neural layers of the experimented deep models of our study which learn high-level features that can then lead to less biased results when they are evaluated on the testing knife set.

2.4. Handling Class Imbalance

To address the overfitting issue in EfficientNet-B0, B6 and DenseNet264, since they excessively learned from the training data, including its noise and anomalies; we implemented techniques such as regularisation, dropout, early stopping, and regularisation, particularly L1 and L2. By doing this, it helps to penalise the complexity of the models, discouraging them from fitting too closely to the training data's noise. Dropout, by randomly disabling neurons during training, prevented the models from becoming overly reliant on any single feature. Early stopping involved halting the training process once our models' performance on a validation set cease to improve, thus avoiding the overtraining that leads to overfitting.

This was accomplished using weighted class predictions embedded into Cross Entropy loss function to resolve imbalanced data classification.

2.5. Custom-Designed Neural Network

Our implementation involved the refinement of the "deit_tiny_patch16_224" DeiT model, for constructing our own designated transformer, OurKnifeNet model. Through meticulous training and adaptation, we achieved a mean Average Precision (mAP) value of 0.7316. The model's initial structure was adept at processing images of 224x224 pixels and incorporated a unique distillation token within its self-attention layers [10], characterised by a stride of 1 and 3x3 kernels. The extraction of a class token from the final Transformer layer was pivotal for image classification.

The pivotal modification in our custom-designed neural network, was the integration of an additional fully connected (fc) layer. This layer extends OurKnifeNet's output dimensionality from 1000 to 192, significantly enhancing its capacity for fine-grained knife recognition. Beyond this addition, the network was augmented with increased Transformer block depth for more effective feature extraction. Dropout layers were carefully placed to prevent overfitting, and an adaptive learning rate schedule

of 0.00005 was implemented to enable fine-tuning of model weights during training.

3. The Proposed Knife Classification System

Based on extensive testing, OurKnifeNet and DenseNet264 models emerged as the optimal architectures for our multi-class classification task. These models were selected after a detailed comparison of various deep Convolutional Neural Networks, as thoroughly discussed in section 5.1, where we outline their superior performance and suitability for our specific requirements.

4. Experimental Analysis of Deep ConvNets

To strengthen our analysis, we present detailed graphs illustrating the variations in validation mean Average Precision (mAP) across different epochs, refer to section 5. This includes a comparative study of both training and validation losses over varying timeframes. Moreover, we employed implicit regularisation techniques, such as applying varying dropout rates to our deep Convolutional Neural Networks (CNNs) and Transformer models.

4.1. Implicit Regularisation – Dropout Rate

We randomly selected neurons to be ignored during training, a technique that we employed in order to prevent overfitting. In particular, the dropout rates being used for our experimental framework were ranging from 0.1 - 0.7 with the latter in OurKnifeNet reaching 0.6831 mAP where the probability of neurons is being dropped.

In some cases, models including EfficientNet-B6, DenseNet264, and OurKnifeNet showed high mAP on training data but a notably lower mAP on testing data, then it typically indicates lack of generalisation, or the model could be too complex for the simplicity of this knife classification, capturing unnecessary details of the training data.

EfficientNet-B6 achieved a peak accuracy of 61.37% when half of the layers were erased, which substantiates the robustness in the network's architecture. DenseNet264 model with a dropout rate of 0.5 has a smaller number of neurons as compared to EfficientNet-B0 with 0.7 dropout rate in fully connected layers, reaching 0.6785 mAP value.

4.2. Label Smoothing

In our knife classification system, we encountered the challenge of noisy labels, a common issue with real-world images of KnifeNet-10k. To mitigate this, we implemented label smoothing within the Cross Entropy Loss function, setting the smoothing value to 0.1. This meant that 10% smoothing was applied to the labels, a strategy designed to regularise our models. By incorporating label smoothing, we aimed to prevent the model from becoming overly confident about its predictions.

4.3. Additive Angular Margin Loss (ArcFace)

Although, this method is primarily used for face recognition, it also holds potential for complex multi-class classification tasks. The rationale for employing this loss function in our experimentation within deep ConvNets and ViT models is to maximise the angular margin between the correct class and the remaining classes of knives. In ArcFace, the core concept involves shifting the similarity measure from Euclidean distance, L2 Norm, to Cosine distance, primarily due to its invariance to partial noise within the data [6].

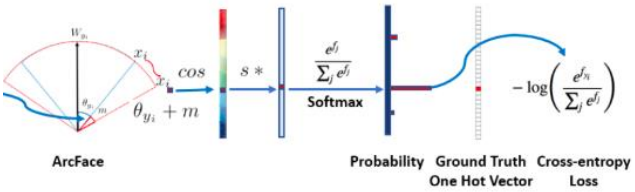


Figure 4: The ArcFace Loss Mechanism

The above schematic illustrates the ArcFace loss function, highlighting the angular margin's role in enhancing feature discriminability by pushing class features away from the decision boundary, thereby improving inter-class separability.

4.4. Boosting Ensemble Mechanisms

Additionally, we integrated a boosting ensemble mechanism, namely XGBoost. This approach enhanced EfficientNet-B0 and DenseNet121 models' predictive accuracy by sequentially correcting errors from previous iterations. Firstly, we trained them and saved their predictions for the training and validation datasets. Then, XGBoost used their predictions as features to make the final classification.

5. Evaluation Metrics

Overall, we evaluated each model by examining their accuracy levels and mAP values. For each model variant, we identified the epochs that yielded the highest validation precision and then re-executed these models using the test subset to determine their final mean average precision results. Monitored concurrently the losses during training and validating the models.

Our evaluations on Knife Image classification show that OurKnifeNet and DenseNet264 outperform by a large margin EfficientNet-B0 and EfficientNet-B6. The test

dataset is used to assess the performance of each model after it has been trained through mAP values. Evaluating how well the models generalise to new and unseen knife images was a crucial concern. We aimed to provide an unbiased evaluation of the models fit during the training phase while validating knife instances. Then, all the deep models were tested on the test set of knives to assess their final performances and eventually select the model variant with the most satisfactory one.

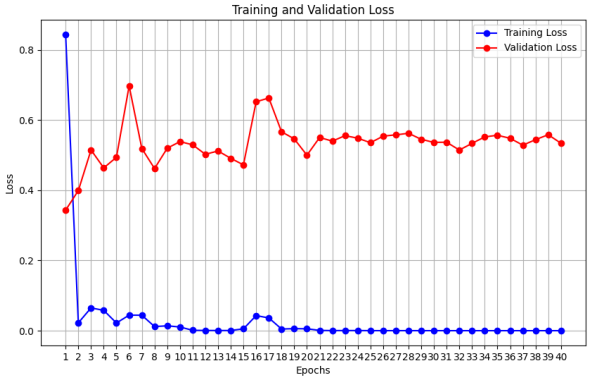
5.1. Accuracy, Precision, Recall

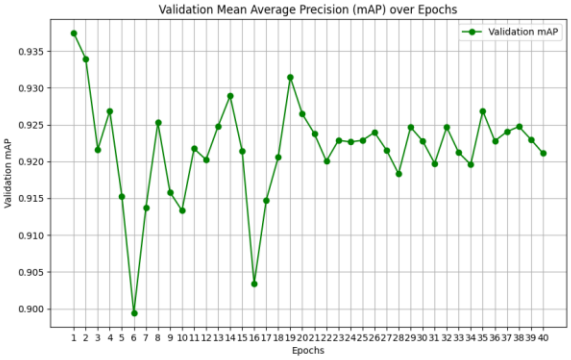
Inferring conclusions based on training and validation loss, and final mAP values during validating and then testing. Our findings are demonstrated within the following table:

Model	PARAMs (M)	FLOPs (B)	Train Loss	Val Loss	Val mAP	Test mAP
EfficientNet-B0	5.3	0.39	0.007	0.417	0.915	0.5818
EfficientNet-B6	43	19	0.002	0.554	0.924	0.6231
DenseNet121	8	2.88	0.000	0.568	0.917	0.6141
DenseNet264	33	6	0.001	0.032	0.926	0.6785
OurKnifeNet	88	17	0.004	0.415	0.932	0.7316

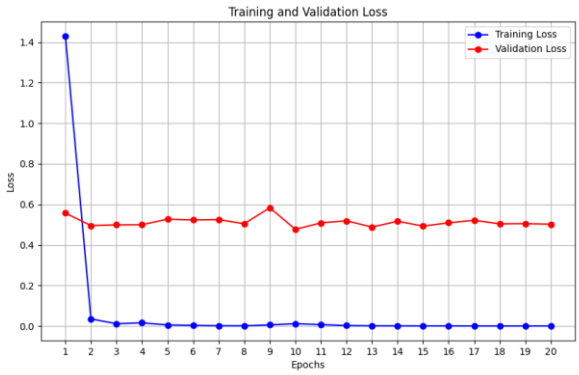
Figure 5: Comparison of Refined Model Variants Performance

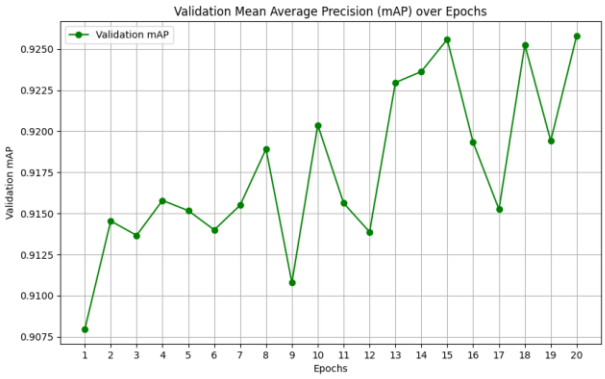
Observation 1: For EfficientNet-B6, a gradual reduction in training and validation loss with a test mAP of 0.6231. Surprisingly, EfficientNet-B6 - Steadily drop in training loss reaching close to 0.006 with 48 samples per batch to load, a learning rate of 0.0005 and 40 epochs.



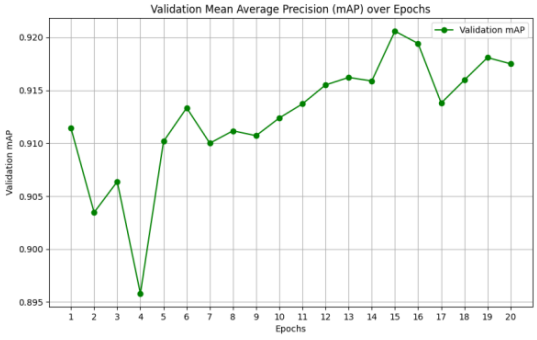


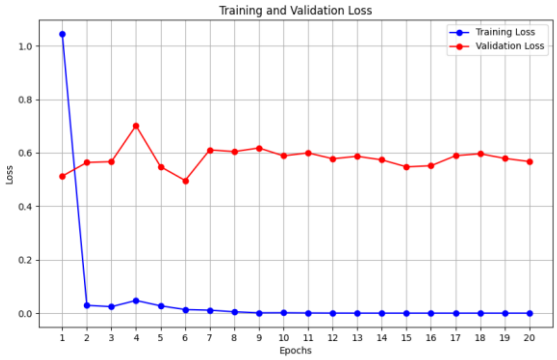
Observation 2: For OurKnifeNet, a gradual reduction in validation loss was recorded with a test mAP of 0.7316.

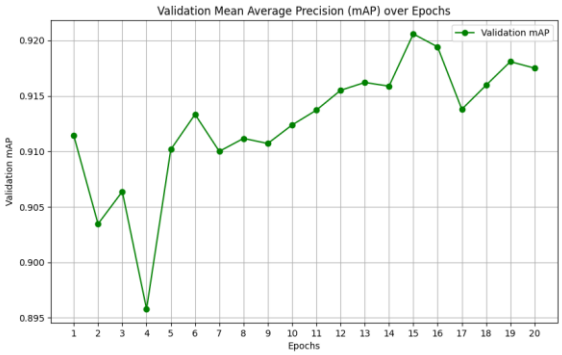




Observation 3: Slight rises were detected in validation loss during training B6 and DenseNet264 which we attributed to the use of a relatively high learning rate. These oscillations were expected, because modifying the models caused the misclassification of a group of knives at a time and thus significantly lowering or raising the mAP.







Finally, we observed that when training the valid loss was decreasing during the first 5 epochs for the baseline model B0 of EfficientNet, then slightly increasing, and gradually reduction close to 0.

5.2. F1 Scores

The F1 score is the harmonic mean of precision and recall, providing a balance between them. It ranges from 0 to 1, where 1 is the best possible score, indicating perfect precision and recall, and 0 is the worst.

5.3. Model Convergence and Divergence

Indeed, a model reaches convergence when it achieves a state during training in which loss settles to within an error range around the final value. That occurred when EfficientNet-B6, DenseNet121 and OurKnifeNet were trained based on the following configurations:

On the whole, additional training will not improve the performance and hence that is the stage we performed early stopping. As for the case of divergence, DenseNet264 failed to ameliorate over iterations and its performance worsens during training drastically due to high learning rate that we set, causing the model to overshoot the minimum of the loss function.

5.4. Macro Average

5.5. Weighted Average

Trained them in a way that they become discriminative CNNs models that distinguish between different classes of knives. Subsequently, we adjusted the weights and biases in the network through training so that the network can accurately identify and then classify different knife inputs.

5.6. Classification Results on KnifeNet-10k

Show sample images & their classification results.

- Multi-Class Confusion Matrix

We evaluate DenseNet-BC with different depths and growth rates on the ImageNet classification task, and compare it with state-of-the-art ResNet architectures. To ensure a fair comparison between the two architectures, we eliminate all other factors such as differences in data preprocessing and optimization settings by adopting the publicly available Torch implementation for ResNet by

5.7. Probability Calibration in Deep CNNs

Considering the prowess of deep convolutional networks in object-recognition tasks from images and yield probabilities for each class in classification operations, we directed our attention towards on calibration curves. We assessed how well the predicted probabilities made by these neural networks align with the actual outcomes. A calibrated classifier from metrics can be evaluated through metrics derived from these reliability diagrams.

6. Future Directions

Our future endeavors will be centered around model convolutional neural network scaling for comparable speedups on memory-limited hardware, i.e., GPU, TPU. Prospective research could explore Stochastic Gradient Descent with Warm Restarts for enhanced training efficiency along with the significance of Test Time Augmentation and Model Calibration. The latter suggests the Expected Calibration Error (ECE) and Brier Score as metrics to evaluate how well the predicted probabilities of a model reflect the true likelihood of outcomes and assess the accuracy of predicted probabilities as the mean squared difference between the predicted probability assigned to the possible outcomes and the actual outcome, respectively.

6.1. Stochastic Gradient Descent with Warm Restarts (SGDR)

In alignment with these findings, it has been proposed in SGDR to employ restart techniques for coordinating the multimodal functions in gradient-free optimization. Accordingly, we could instantiate a warm restart technique for SGD to improve its anytime performance when training DNNs. That is, the partial warm restarts in gradient-based optimisation are being executed in order to enhance the convergence rate in accelerated gradient schemes, specifically for addressing ill-conditioned functions [11].

6.2. Scaling up Models for ConvNets

An approach is to scale a Baseline Model with Different Network Width (w), Depth (d), and Resolution (r) Coefficients, "which can result in about $O(s)$ increase in model activation scaling flops by a factor of s, the proposed fast compound scaling results in close to $O(\sqrt{s})$ increase in activations, while achieving excellent accuracy [12][14]. "

6.3. Test Time Augmentation (TTA)

Data augmentation during the testing phase holds potential for enhancing the performance of the trained networks. TTA will allow our pre-trained models to make predictions on multiple augmented versions of each test knife image, thereby increasing their robustness to variations in new data. By averaging the combined predictions from these different versions of images, we anticipate a reduction in prediction errors [13], particularly in cases where test data differs from training data due to distortions and variations in the test knife images.

Model	top-1	top-5
DenseNet-121	25.02 / 23.61	7.71 / 6.66
DenseNet-169	23.80 / 22.08	6.85 / 5.92
DenseNet-201	22.58 / 21.46	6.34 / 5.54
DenseNet-264	22.15 / 20.80	6.12 / 5.29

Table 3: The top-1 and top-5 error rates on the ImageNet validation set, with single-crop / 10-crop testing.

We select the model with the lowest validation error during training and report the test error.

Add for Data Augmentation:
We follow [42] and divide the pixel values by 255 so they are in the [0, 1] range. We adopt the same data augmentation scheme for training images as in [], and apply a single-crop or 10-crop with size 224×224 at test time. Following [], we report classification errors on the validation set.

Add section:

Training

All the networks are trained using stochastic gradient descent (SGD). On CIFAR and SVHN we train using batch size 64 for 300 and 40 epochs, respectively. The initial learning rate is set to 0.1, and is divided by 10 at 50% and 75% of the total number of training epochs. On ImageNet, we train models for 90 epochs with a batch size of 256. The learning rate is set to 0.1 initially, and is lowered by 10 times at epoch 30 and 60. Note that a naive implementation of DenseNet may contain memory inefficiencies. To reduce the memory consumption on GPUs, please refer to our technical report on the memory-efficient implementation of DenseNets [26].

Following [8], we use a weight decay of 10−4 and a Nesterov momentum [35] of 0.9 without dampening. We adopt the weight initialization introduced by [10]. For the three datasets without data augmentation, i.e., C10, C100 and SVHN, we add a dropout layer [33] after each convolutional layer (except the first one) and set the dropout rate to 0.2. The test errors were only evaluated once for each task and model setting.

Perturbation Methods

References

[1] Zagoruyko, S. and Komodakis, N. Wide residual networks. BMVC, 2016

[2] H2O.ai, n.d. Weights and Biases. Available at: https://h2o.ai/wiki/weights-and-biases/

[3] *Li, H., Chaudhari, P., Yang, H., Lam, M., Ravichandran, A., Bhotika, R. & Soatto, S., 2020. Rethinking the Hyperparameters for Fine-tuning. CoRR,*

[4] Hu, J., Shen, L. & Sun, G., 2017. Squeeze-and-Excitation Networks. *CoRR,*

[5] Papers With Code, n.d. Weight Decay Explained.

[6] Deng, J., Guo, J. & Zafeiriou, S., 2018. ArcFace: Additive Angular Margin Loss for Deep Face Recognition. *CoRR,*

[7] Huang, G., Liu, Z. & Weinberger, K.Q., 2016. Densely Connected Convolutional Networks, CoRR.

[8] S X. -S. Wei *et al*., "Fine-Grained Image Analysis With Deep Learning: A Survey," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 12, pp. 8927-8948, 1 Dec. 2022, doi: 10.1109/TPAMI.2021.3126648.

[9] Hugging Face, n.d. DeiT (Data-efficient Image Transformers) Documentation. Available at: https://huggingface.co/docs/transformers/model_doc/deit

[10] Zhang, A., Lipton, Z.C., Li, M. & Smola, A.J., n.d. Transformer. *Dive into Deep Learning*.

[11] Brock, A., Donahue, J. & Simonyan, K., 2017. Large Scale GAN Training for High Fidelity Natural Image Synthesis.

[12] Tan, M. & Le, Q.V., 2019. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. *CoRR*.

[13] Zhang, M., Levine, S. & Finn, C., 2021. MEMO: Test Time Robustness via Adaptation and Augmentation. *CoRR*.

[14] Dollár, P., Singh, M. & Girshick, R.B., 2021. Fast and Accurate Model Scaling. *CoRR*.

7. Appendix

All below are for DenseNets[]:

Stochastic vs. deterministic connection: There is an interesting connection between dense convolutional networks and stochastic depth regularization of residual networks [13]. In stochastic depth, layers in residual networks are randomly dropped, which creates direct connections between the surrounding layers. As the pooling layers are never dropped, the network results in a similar connectivity pattern as DenseNet: there is a small probability for any two layers, between the same pooling layers, to be directly connected—if all intermediate layers are randomly dropped. Although the methods are ultimately quite different, the DenseNet interpretation of stochastic depth may provide insights into the success of this regularizer.

Feature Reuse: By design, DenseNets allow layers access to feature-maps from all of its preceding layers (although sometimes through transition layers). We conduct an experiment to investigate if a trained network takes advantage of this opportunity. We first train a DenseNet on C10+ with L = 40 and k = 12. For each convolutional layer ` within a block, we compute the average (absolute) weight assigned to connections with layer s. Figure 5 shows a heat-map for all three dense blocks. The average absolute weight serves as a surrogate for the dependency of a convolutional layer on its preceding layers. A red dot in position (`, s) indicates that the layer ` makes, on average, strong use of feature-maps produced s-layers before. Several observations can be made from the plot:

1. All layers spread their weights over many inputs within the same block. This indicates that features extracted by very early layers are, indeed, directly used by deep layers throughout the same dense block.
2. The weights of the transition layers also spread their weight across all layers within the preceding dense block, indicating information flow from the first to the last layers of the DenseNet through few indirections.
3. The layers within the second and third dense block consistently assign the least weight to the outputs of the transition layer (the top row of the triangles), indicating that the transition layer outputs many redundant features (with low weight on average). This is in keeping with the strong results of DenseNet-BC where exactly these outputs are compressed.
4. Although the final classification layer, shown on the very right, also uses weights across the entire dense block, there seems to be a concentration towards final feature-maps, suggesting that there may be some more high-level features produced late in the network.

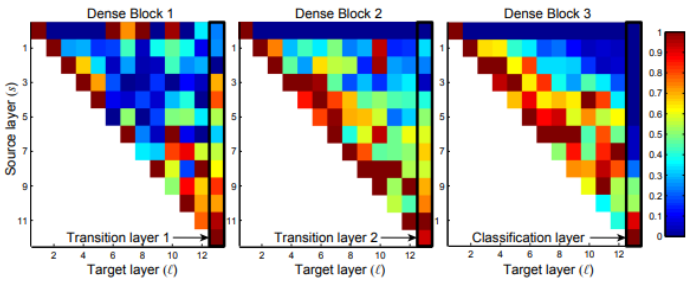


Figure 5: The average absolute filter weights of convolutional layers in a trained DenseNet. The color of pixel (s, `) encodes the average L1 norm (normalized by number of input feature-maps) of the weights connecting convolutional layer s to ` within a dense block. Three columns highlighted by black rectangles correspond to two transition layers and the classification layer. The first row encodes weights connected to the input layer of the dense block.

Conclusion

We showed that DenseNets scale naturally to hundreds of layers, while exhibiting no optimization difficulties. In our experiments, DenseNets tend to yield consistent improvement in accuracy with growing number of parameters, without any signs of performance degradation or overfitting.

KL Divergence: