NNMOBILE-NET

Code for the paper nnMobileNet: Rethinking CNN for Retinopathy Research

This repository will be maintained and updated! Stay Tuned!

We will appreciate any suggestions and comments. If you find this code helpful, please cite our papers. Thanks!

```
@inproceedings{zhu2024nnmobilenet,
  title={nnMobileNet: Rethinking CNN for Retinopathy Research},
  author={Zhu, Wenhui and Qiu, Peijie and Chen, Xiwen and Li, Xin and Lepore, Natasha and
  booktitle={Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recogni
  pages={2285--2294},
  year={2024}
}
```


Abstract

Over the past few decades, convolutional neural networks (CNNs) have been at the forefront of the detection and tracking of various retinal diseases (RD). Despite their success, the emergence of vision transformers (ViT) in the 2020s has shifted the trajectory of RD model development. The leading-edge performance of ViT-based models in RD can be largely credited to their scalability—their ability to improve as more parameters are added. As a result, ViT-based models tend to outshine traditional CNNs in RD applications, albeit at the cost of increased data and computational demands. ViTs also differ from CNNs in their approach to processing images, working with patches rather than local regions, which can complicate the precise identification of small, variably presented lesions in RD. In our study, we revisited and updated the architecture of a CNN model, specifically MobileNet, to enhance its utility in RD diagnostics. We found that an optimized MobileNet, through selective modifications, can surpass ViT-based models in various RD benchmarks, including diabetic retinopathy grading, detection of multiple fundus diseases, and classification of diabetic macular edema.

Experiments

Table 2: Comparison of rDR and normal classification on the Messidor-1 dataset [21]. Annotations denote whether pixel-level or patch-level supervision was applied.

Method	Annotations	Referral AUC	Normal AUC
VNXK [33]	-	88.7	87.0
CKML [33]	-	89.1	86.2
Comp. CAD [6]	-	91.0	87.6
Expert A [6]	-	94.0	92.2
Expert B [6]	-	92.0	86.5
Zoom-in-Net [3]	-	95.7	92.1
AFN [5]	patch	96.8	-
Semi + Adv [4]	pixel	97.6	94.3
CANet[2]	-	96.3	-
LAT [10]	-	98.7	96.3
Ours	-	**98.7**	**97.5**

Table 3: Performance comparison of multi-disease abnormal detection on the RFMiD dataset [29]. Param are the parameter numbers, indicating model complexity of models.

		Normal/Abnormal		
Method	Param(M)	ACC	AUC	F1
CANet[2]	29	88.3	91.0	90.4
EffNet-B7[34]	66	88.2	91.0	90.7
ReXNet[17]	34	91.3	94.5	93.3
CrossFormer-L[35]	92	90.6	94.3	92.0
Swin-L[36]	197	89.5	93.8	91.6
MIL-VT[9]	98	91.1	95.9	94.4
SatFormer-B [11]	78	93.8	96.5	**95.8**
Ours	34	**94.4**	**98.7**	94.4

Table 4: Performance comparison of DR grading on the APOTS dataset [30].

	DR Grading			
Method	AUC	ACC	F1	Kappa
DLI[37]	-	82.5	80.3	89.0
CANet[2]	-	83.2	81.3	90.0
GREEN-ResNet50 [38]	-	84.4	83.6	90.8
GREEN-SE-ResNext50 [38]	-	85.7	85.2	91.2
MIL-VT [9]	**97.9**	85.5	85.3	92.0
Ours	97.8	**89.1**	**88.9**	**93.4**

Table 5: Performance comparison of DR and DME grading on the IDRID dataset [31].

Method	DME AUC	DME F1	DME ACC	DR AUC	DR F1	DR ACC
CANet[2]	87.9	66.1	78.6	78.9	42.3	57.3
Multi-task net[39]	86.1	60.3	74.8	78	43.9	59.2
MTMR-net[40]	84.2	61.1	79.6	79.7	45.3	60.2
DETACH + DAW [7]	89.5	72.3	82.5	84.8	49.4	59.2
Ours	**95.3**	**84.8**	**86.5**	**91.6**	**72.6**	**73.1**

We also brought this model to MICCAI MMAC 2023, and won the 3rd.

Table 6. MMAC official released the final ranking.

Method	Kappa	F1	SPE	Average	CPU time (s)
Rank 1^{st}	90.1	78.1	94.5	87.5	2.1283
Rank 2^{nd}	88.9	76.8	94.1	86.6	0.8047
Ours (Rank 3^{rd})	90.0	75.1	94.1	86.4	0.2750

```
------------------------------------------------------------------------------------------
In main.py -> ImageNet pretrained weight
    model = ReXNetV1(width_mult=3.0,classes=args.nb_classes,dropout_path=args.drop_path)
    model.load_state_dict(torch.load('rexnet_3.0.pth'),strict=False)
------------------------------------------------------------------------------------------
```

rexnet_3.0.pth download from Google Drive Or Rexnet Official repo

Single-GPU

```
python main.py  --input_size 224 --drop_path 0.2
```

Multi-GPU

```
python -m torch.distributed.launch --nproc_per_node=4 main.py --batch_size 24 --lr 1e-3 --
```

License

Released under the ASU GitHub Project License.