How to train your ViT for OOD detection

Maximilian Müller and Matthias Hein
University of Tübingen and Tübingen AI Center
maximilian.mueller@wsii.uni-tuebingen.de

Abstract

VisionTransformers have been shown to be powerful out-of-distribution detectors for ImageNet-scale settings when finetuned from publicly available checkpoints, often outperforming other model types on popular benchmarks. In this work, we investigate the impact of both the pretraining and finetuning scheme on the performance of ViTs on this task by analyzing a large pool of models. We find that the exact type of pretraining has a strong impact on which method works well and on OOD detection performance in general. We further show that certain training schemes might only be effective for a specific type of out-distribution, but not in general, and identify a best-practice training recipe.

1 Introduction

Deep neural networks have undeniably achieved remarkable success across a spectrum of real-world applications, showcasing outstanding performance. Nevertheless, they often exhibit unforeseen behaviour when confronted with unknown situations like receiving an input that is not related to the task it has been trained on. These anomalous samples are commonly known as out-of-distribution (OOD) data, and deep neural network classifiers often make unwarrantedly confident predictions, incorrectly categorizing them within the realm of in-distribution (ID) classes (Hendrycks and Gimpel, 2017; Hein et al., 2019).

In response to this challenge, the development of OOD detectors has emerged as a research field. These detectors serve the purpose of identifying OOD inputs, while at the same time letting ID inputs pass through, such that depending on the application appropriate measures can be taken. Recent research has shown that VisionTransformers can perform particularily well on this task. An investigation by Galil et al. (2023) found that "a subset of ViTs achieves the best [...] OOD detection performance" in their benchmark under various training regimes. However, not all training strategies yielded competitive results. Further, OOD detection methods that directly access pre-logit feature information, such as the Mahalanobis distance Lee et al. (2018), were excluded from their study, presumably due to the increased computational cost arising from the necessity of iterating over the ImageNet (Deng et al., 2009) train set for feature extraction. However, Bitterwolf et al. (2023), who also identified certain ViTs as the top performers in their benchmark, reported that precisely these feature-based methods exhibit the highest efficacy, particularly when coupled with supervised pretraining on the ImageNet-21k dataset. They further hypothesize that the choice of pretraining might have a profound impact on OOD detection performance.

Motivated by these findings, the primary objective of this paper is to undertake an extensive exploration of the role played by pretraining and, more broadly, training hyperparameters for ViTs in the context of OOD detection.

2 Experimental Setup

In order to investigate the role of pretraining and hyperparameters on large-scale OOD detection in a controlled setting, we combine two publicly available model pools and evaluate them with commonly used post-hoc OOD detection methods. We report FPR values in the main paper, but AUC values can be found in the Appendix and lead to the same conclusions.

Models. We evaluate ViT-B/16-224 and ViT-B/32-224 weights from Steiner et al. (2022), which were trained from scratch on ImageNet-1k or ImageNet-21k with varying hyperparameter settings

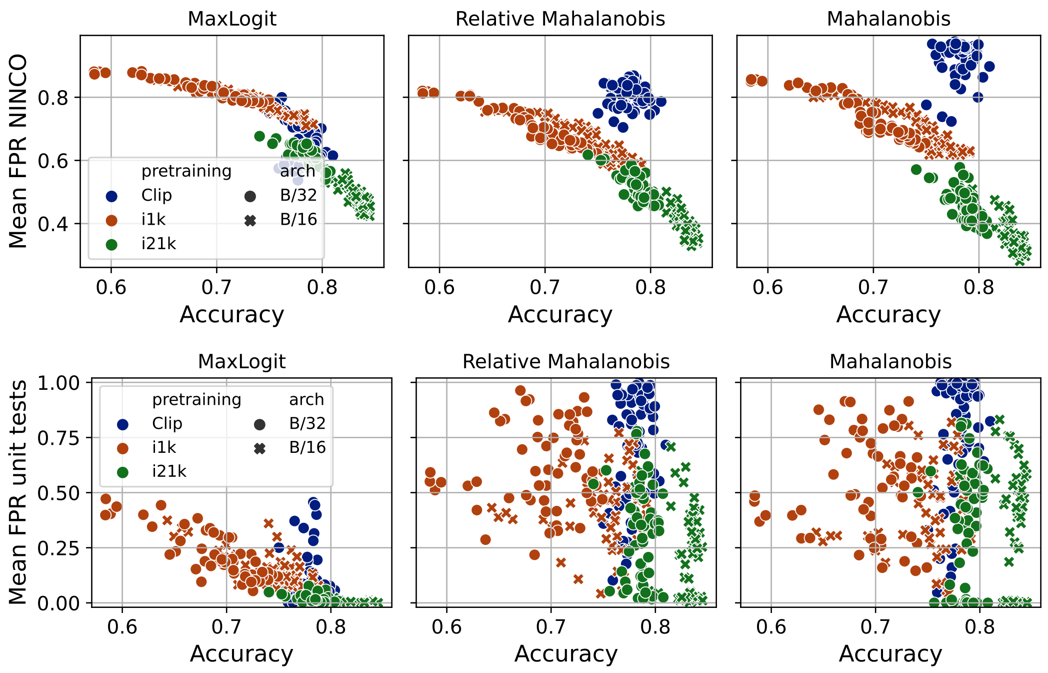

Figure 1: **Pretraining matters:** ImageNet-21k pretraining paired with Mahalanobis-based detection methods strongly outperforms other detectors on NINCO, yet fails in many cases on the unit-test task.

(including augmentations, weight decay, and stochastic depth / dropout) in a supervised fashion, and subsequently finetuned on ImageNet-1k for 20.000 steps with 2 different learning rates, leading to a rich pool of 222 models. We further evaluate models from Wortsman et al. (2022), who investigated the benefits of "souping" models that were finetuned from a common checkpoint with different hyperparameters. To this end, they finetuned a CLIP-pretrained ViT-B/32-224 on ImageNet-1k with different settings (varying augmentations, mixup, weight decay, lr, finetuning epochs and smoothing). Wortsman et al. (2022) eventually combine them either in a uniform or greedy fashion for enhanced performance, and we evaluate both the individual and the souped models. Overall, our pool comprises 296 different models (details in Appendix A).

Dataset. In a recent study Bitterwolf et al. (2023) showed that most of the commonly used datasets for OOD detection evaluation for ImageNet-1k are severely flawed and contain a significant fraction of ID samples. They proposed the hand-cleaned *NINCO* dataset as a solution, which we will therefore use for our evaluations. *NINCO* consists of real-world images, with several 'hard' (i.e. similar to ImageNet-1k) classes, hence mimicking an OOD test distribution which is in parts very close to the in-distribution. Further, they provide a range of synthetic *unit-test* sets, mimicking noise distributions that might occur in the real world, but should be 'far' from the ImageNet test set. A good detector should be able to solve both tasks, and we will therefore contrast the results on *NINCO* with those on the *unit-tests*.

Methods. Post-hoc OOD detection methods can be separated into methods that directly access pre-logit feature information (called **feature-based**) and methods that only leverage logit or softmax information (called **logit-based**). In the main paper, we show results for the respective best methods: Max-Logit (Hendrycks et al., 2022), Mahalanobis (Lee et al., 2018) and relative Mahalanobis distance (Ren et al., 2021). We provide results for more methods in Appendix B.4 and give an overview over all methods in Appendix C.

3 Observations

Pretraining Dataset. In Figure 1 we plot the mean FPR of all investigated models against their in-distribution accuracy for three methods and the two OOD test sets. For Max-Logit, the FPR on NINCO (first row) correlates fairly well with accuracy, irrespective of the pretraining dataset. For the feature-dependent Mahalanobis-based methods, the models with CLIP-pretraining perform significantly worse than models with ImageNet-21k or ImageNet-1k pretraining, even for similar ID accuracy. Those feature-based methods, however, lead to the lowest overall FPR values on NINCO observed in our study when combined with ImageNet-21k pretraining. When considering the supposedly easy unit-test dataset (second row in Figure 1), the picture changes drastically: Especially for Mahalanobis-based methods the correlation between accuracy and FPR vanishes, and many models with similar test accuracy show vastly different FPR values. Some of the strongest models in terms of accuracy show very poor OOD detection capabilities, often with FPR values over 50%. This is concerning, since it indicates that supposedly strong OOD detectors, i.e. ones that are capable of solving the difficult task of separating near-OOD data from ID data, are vulnerable to OOD data from very different distributions. Overall, the strongest OOD detectors are clearly obtained from ImageNet-21k pretraining both for NINCO and the unit-tests, and in many cases CLIP-pretraining can at most slightly improve over ImageNet-1k-only training. In the following, we will go beyond the effects of the pretraining dataset and investigate the role of specific training parameters for each pretraining dataset.

Pretraining on ImageNet-21k and finetuning on ImageNet-1k. When pairing models with ImageNet-21k pretraining (Figure 3) with logit-based methods like MaxLogit, all models yield FPR values close to 0 for the unit-tests, basically solving the task. There is however a difference on the NINCO dataset, where the FPR values range roughly between 45% and 60%. The better models are usually finetuned with larger learning rate, and were pretrained without dropout and with smaller weight decay. Augmentations seem to play less of a role. While the worst models in terms of FPR also show worse accuracy, there are also high-accuracy models with high FPR. Those were typically trained with strong augmentations and dropout and finetuned with small learning rate.

When pairing the models with Mahalanobis distance as OOD detection method, the overall best results on the NINCO dataset are achieved, with the best FPR values below 30%. There are however strong differences between the specific detectors when considering the unit tests. While models that were pretrained with larger weight decay and finetuned with smaller learning rate lead to detectors that perform well on both NINCO and the unit-tests, models with smaller weight decay often and with larger learning rate always fail on the unit tests, with FPR values between 20% and 80%. Concerningly, those models showed some of the highest test accuracies (>84.0%) in our study. On the contrary, there are models with relatively low test accuracy (e.g. 79%), but close-to-zero FPR on the unit tests. When conditioning on larger weight decay and smaller learning rate, the FPR on NINCO correlates well with pretraining validation accuracy ($\tau = -0.72$), which provides a best-practice training strategy: Selecting models that were pretrained on ImageNet-21k with larger weight decay, and amongst those the ones with highest pretraining validation accuracy, and then finetune them with small learning rate (more in App. B.2).

Training exclusively on ImageNet-1k. When pairing models trained only on ImageNet-1k (Figure 2) with MaxLogit, there is a strong correlation between the performance on NINCO and the test accuracy ($\tau = 0.88$), i.e. settings that improve the in-distribution accuracy (e.g. stronger augmentations, dropout and to a smaller extent also stronger weight decay) also tend to improve the OOD detection capabilities for NINCO. While the correlation for the unit tests is smaller, stronger augmentations still lead to better performance.

When pairing models trained only on ImageNet-1k with Mahalanobis, lower FPR values on NINCO can be achieved compared to logit-based methods. Stronger augmentations lead to stronger performance on NINCO (and also higher test accuracy). However, there are many cases where the models completely fail on the unit tests (FPR values >65%), even when pretrained with strong augmentations. Some of those are amongst the best-performing models in terms of test accuracy (when pretrained with dropout, small weight decay and finetuned with large learning rate). The best detectors on the unit tests were pretrained with larger weight decay and finetuned with smaller learning rate, and there are only relatively few models that perfrom well on both NINCO and the unit-tests. For ImageNet-1k pretraining the *Relative* Mahalanobis distance works slightly better than the conventional Mahalanobis distance, but the overall trends remain similar.

Pretraining on CLIP and finetuning on ImageNet-1k. For models pretrained with CLIP (Figure 5 in the appendix), feature-based methods like Mahalanobis distance (but also other methods like ViM as shown in appendix B.4) bring no improvements over logit-based methods, with FPR values commonly above 90%. The best method for those models is MaxLogit, but other logit-based methods (e.g. Energy, KL-Matching, ...) perform similar. In general, accuracy does not correlate

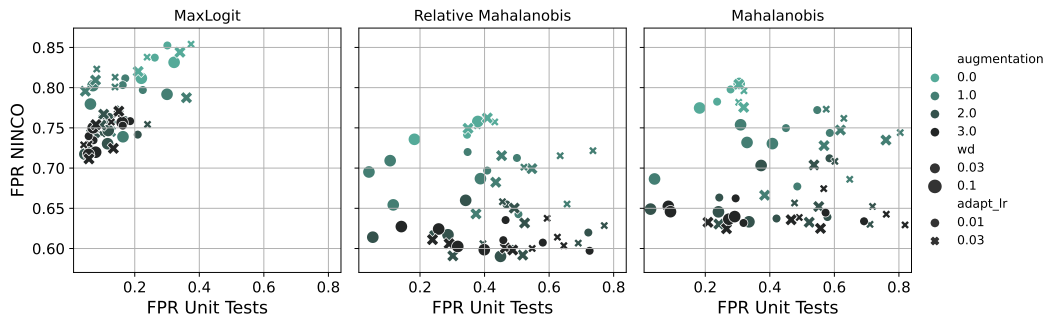

Figure 2: ViT-B/16 trained exclusively on ImageNet-1k

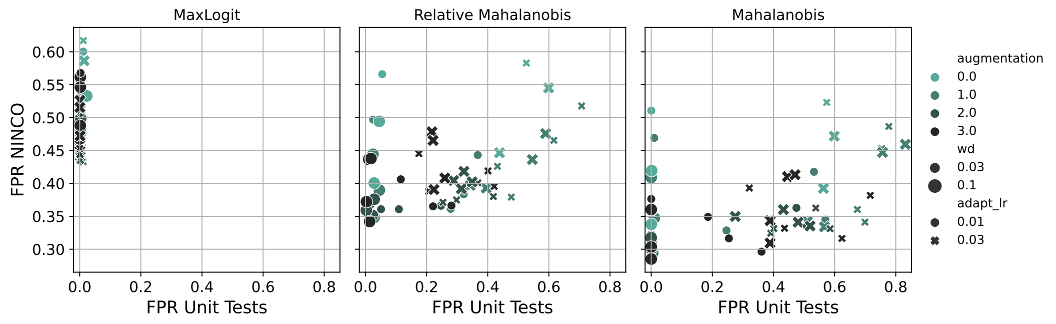

Figure 3: ViT-B/16 with pretraining on ImageNet-21k

well with FPR, and often there is no significant difference in the OOD detection capabilites of models with fairly different test accuracies. The souped models bring no consistent advantage over the individual models. For MaxLogit, many models achieve 0% FPR on the unit tests, and all models with strong rand-aug perform relatively well (even though also models without strong rand-aug *can* achieve 0% FPR).

Do we need finetuning at all? Due to the strong performance of ViTs with ImageNet-21k pretraining it is natural to ask whether training *only* on ImageNet-21k might be enough to obtain strong feature-based detectors for ImageNet-1k. In Table 3 in App. B.2 we report the performance of the 21k-checkpoints with Mahalanobis distance *before* finetuning and show that this is not the case: Finetuning improves the detection performance consistently. We provide an extended discussion on this in App. B.2.

4 Conclusions

In this study we showed that for ViTs the effectiveness of an OOD detection method depends crucially on the detailed pretraining and finetuning scheme of the model. While pretraining on ImageNet-21k yields clearly the best detectors (as also observed for generalization tasks by Steiner et al. (2022)), many detectors, especially feature-based ones, are susceptible to small changes in the hyperparameters and might only work for a particular type of out-distribution.

In their work, Steiner et al. (2022) recommended choosing the ImageNet-21k pretrained checkpoints with highest upstream validation accuracy for finetuning on downstream tasks. We show that this is not enough for OOD detection, since e.g. a too large learning rate can result in poorly performing Mahalanobis-based detectors, even though the validation accuracy both before and after finetuning might be comparably high (more details can be found in App B.2).

Consistently good results were obtained for Mahalanobis distance as OOD detection method with models pretrained on ImageNet-21k with larger weight decay during pretraining and subsequent finetuning with small learning rate. We hypothesize that larger weight decay might help to structure the feature space in a beneficial way during supervised pretraining, and that a smaller finetuning learning rate might preserve this structure better. CLIP pretraining, which is based on a cosine-similarity measure, seems to impose a different structure, which does not lend itself towards the current feature-based OOD detection methods.

Acknowledgments

We acknowledge support from the German Federal Ministry of Education and Research (BMBF) through the Tübingen AI Center (FKZ: 01IS18039A) and from the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy (EXC number 2064/1, Project number 390727645), as well as from the Carl Zeiss Foundation in the project "Certification and Foundations of Safe Machine Learning Systems in Healthcare" and the European Laboratory for Learning and Intelligent Systems (ELLIS).

References

J. Bitterwolf, M. Mueller, and M. Hein. In or out? fixing imagenet out-of-distribution detection evaluation. In *ICML*, 2023. URL https://proceedings.mlr.press/v202/bitterwolf23a.html.

J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

I. Galil, M. Dabbah, and R. El-Yaniv. A framework for benchmarking class-out-of-distribution detection and its application to imagenet. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=Iuubb9W6Jtk.

M. Hein, M. Andriushchenko, and J. Bitterwolf. Why ReLU networks yield high-confidence predictions far away from the training data and how to mitigate the problem. In *CVPR*, 2019.

D. Hendrycks and K. Gimpel. A baseline for detecting misclassified and out-of-distribution examples in neural networks. In *ICLR*, 2017. URL https://openreview.net/forum?id=Hkg4TI9xl.

D. Hendrycks, S. Basart, M. Mazeika, A. Zou, J. Kwon, M. Mostajabi, J. Steinhardt, and D. Song. Scaling out-of-distribution detection for real-world settings. In *ICML*, 2022.

K. Lee, K. Lee, H. Lee, and J. Shin. A simple unified framework for detecting out-of-distribution samples and adversarial attacks. In *NeurIPS*, 2018.

W. Liu, X. Wang, J. Owens, and Y. Li. Energy-based out-of-distribution detection. *Advances in Neural Information Processing Systems*, 2020.

J. Ren, S. Fort, J. Liu, A. G. Roy, S. Padhy, and B. Lakshminarayanan. A simple fix to mahalanobis distance for improving near-ood detection, 2021. URL https://arxiv.org/abs/2106.09022.

A. P. Steiner, A. Kolesnikov, X. Zhai, R. Wightman, J. Uszkoreit, and L. Beyer. How to train your vit? data, augmentation, and regularization in vision transformers. *TMLR*, 2022. URL https://openreview.net/forum?id=4nPswr1KcP.

Y. Sun, C. Guo, and Y. Li. React: Out-of-distribution detection with rectified activations. *NeurIPS*, 2021.

Y. Sun, Y. Ming, X. Zhu, and Y. Li. Out-of-distribution detection with deep nearest neighbors. *ICML*, 2022.

J. Tack, S. Mo, J. Jeong, and J. Shin. Csi: Novelty detection via contrastive learning on distributionally shifted instances. In *NeurIPS*, 2020.

H. Wang, Z. Li, L. Feng, and W. Zhang. Vim: Out-of-distribution with virtual-logit matching. In *CVPR*, 2022.

M. Wortsman, G. Ilharco, S. Y. Gadre, R. Roelofs, R. Gontijo-Lopes, A. S. Morcos, H. Namkoong, A. Farhadi, Y. Carmon, S. Kornblith, and L. Schmidt. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *ICML*, 2022. URL https://proceedings.mlr.press/v162/wortsman22a.html.

J. Yang, P. Wang, D. Zou, Z. Zhou, K. Ding, W. Peng, H. Wang, G. Chen, B. Li, Y. Sun, et al. Openood: Benchmarking generalized out-of-distribution detection. *arXiv preprint arXiv:2210.07242*, 2022.

5 Supplementary Material

A Experimental details

We report the detailed hyperparameter grid for the models from Steiner et al. (2022) in Table 1 and from Wortsman et al. (2022) in Table 2. For the plots in the main paper except for Figure 1, we show only the B/16 variants of the Steiner et al. (2022) models for better visibility, but the observed trends remain the same for the B/32 variants. We further aggregate the augmentations from Table 1 into {none,light, medium, strong}. For CLIP-models (e.g. in Figure 5), we plot the *magnitude* of rand-aug, assume min-aug to have magnitude 0, and discard the number of augmentations. We further aggregate mixup into weak (≤ 0.15) and strong (> 0.15) for easier visualization.

Table 1: Hyperparameter grid for the selected ViTs from Steiner et al. (2022). Hyperparameters vary for both pretraining and finetuning.

	Pretraining						**Finetuning**		
arch	epochs	lr	dset	wd	aug	dropout / stoch. d.	lr	steps	res.
B/16, B/32	300	0.001	IN-1k, IN-21k	0.03, 0.1	strong1,strong2 med1,med2 light0,light1 none	0, 0.1	0.01, 0.03	20k	224

Table 2: Hyperparameter grid for the CLIP-pretrained ViTs from Wortsman et al. (2022). Hyperparameters vary only for finetuning.

	Finetuning							
arch	aug	r-aug n	r-aug m	mixup	smoothing	epochs	wd	lr
B/16	min with prob 1/3 else r-aug	0,1,2	$[0,20]$	0 with prob 0.5, else $[0,0.9]$	0 with prob 0.5, else $[0,0.25]$	[4,16]	$10^{-\lambda}$, $\lambda \in [4,6]$	$10^{-\lambda}$, $\lambda \in [0.2,4]$

B Additional Results

B.1 Pretraining plots with AUC

In Figure 4 we show the mean AUC on NINCO and the unit tests vs. in-distribution accuracy to complement the FPR plot from the main paper. The conlusions remain the same: The best detectors on NINCO are obtained with Mahalanobis distance and ImageNet-21k pretraining, yet some of those fail completely on the unit tests.

B.2 Detailed results for ViT-B/16 with ImageNet-21k pretraining

We report detailed results for the ViT-B/16 models with Imagenet-21k pretraining in Table 3 (FPR) and Table4 (AUC), since this setup yields the overall strongest detectors. The Table is sorted according to the validation accuracy of the pretraining checkpoints (higher on top), since Steiner et al. (2022) suggest selecting the models with highest pretraining accuracy for finetuning on downstream tasks. While this works relatively well for OOD detection on NINCO (the top models all achieve FPR values around 30% with Mahalanobis distance), the effect of the learning rate on Mahalanobis distance is striking: For most checkpoints, finetuning with a learning rate of 0.01 (white rows) leads to FPR values of around 0% (solving the unit-test task), whereas finetuning with a learning rate of 0.03 (grey rows) never yields FPR values below 30%, and often even much

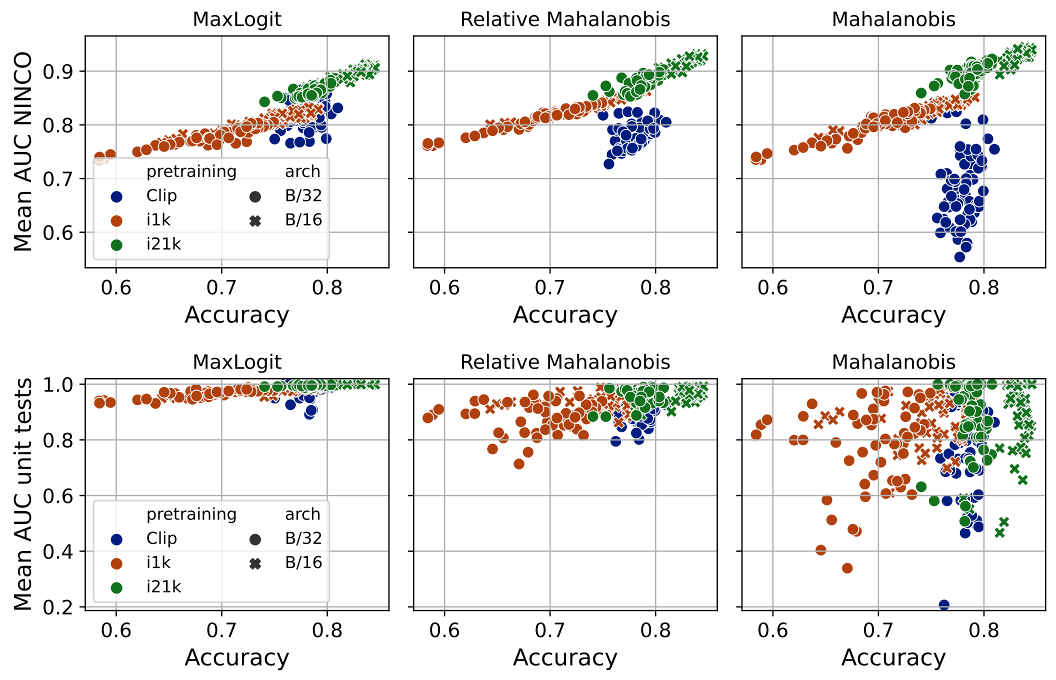

Figure 4: **Pretraining matters:** ImageNet-21k pretraining paired with Mahalanobis-based detection methods strongly outperforms other detectors on NINCO, yet fails in many cases on the unit-test task.

larger ones, outlining that the structure of the feature space that lends itself towards Mahalanobis distance seems to be crucially affected by the learning rate, even though this might not strongly affect the models generalization performance.

In order to achieve good performance on both NINCO and the unit tests, the best strategy is thus to select models that were pretrained on ImageNet-21k with larger weight decay, and amongst those the ones with highest pretraining validation accuracy, and then finetune them with small learning rate and combine them with Mahalanobis distance as detection method. One remaining question is how to achieve strong pretraining accuracy. This is thoroughly discussed in Steiner et al. (2022) and depends non-trivially on the interplay between augmentation, regularization and model capacity. For our study, we observed that good results could be obtained, as long as strong augmentations are not paired with dropout, and none or light augmentations are not paired with zero dropout. Arguably, this might however change when considering different model sizes.

Since the most powerful detectors in our survey are obtained with pretraining on ImageNet-21k, the question arises whether training *only* on ImageNet-21k might be enough to obtain strong detectors for ImageNet-1k. We therefore also report the performance of the pretraining checkpoints with Mahalanobis distance, dubbed *Mahalanobis (pre)* in the table. This is, we evaluate the checkpoint *before* finetuning on ImageNet-1k. Due to the different output dimension of ImageNet-1k and ImageNet-21k models (1k vs. 21k classes), evaluating 21k-checkpoints is only possible for methods that operate exclusively in the feature space (like [relative] Mahalanobis distance, knn and cosine). Notably, the performance of Mahalanobis distance after finetuning is *always* better for the respective checkpoints, indicating that the structure of the feature space adapts significantly during the finetuning procedure. Nevertheless, certain pretraining checkpoints yield remarkably good results on NINCO, with FPR values of roughly 40% - results that are not achieved by any method with CLIP pretraining or ImageNet-1k-exclusive training. On the unit tests, however, the pretraining checkpoints fail consistently, showing FPR values of 40-90%.

B.3 Clip Pretraining

In Figure 5 we show the plots corresponding to the discussion of the CLIP models in the main paper.

Table 3: Detailed hyperparameters and FPR values for the ViT-B/16 models with ImageNet-21k pretraining. In the *Mahalanobis (pre)* columns we evaluate the pretraining checkpoint (i.e. *before* finetuning) with Mahalanobis distance. As suggested by Bitterwolf et al. (2023), we also report the number of failed unit tests (with FPR values ≥ 10%).

						MaxLogit			**Mahalanobis**			**Mahalanobis (pre)**		
						NINCO	unit tests		NINCO	unit tests		NINCO	unit tests	
wd	aug	do	adapt-lr	pre-acc	ft-acc	FPR [%]	FPR [%]	n failed	FPR [%]	FPR [%]	n failed	FPR [%]	FPR [%]	n failed
0.1	medium1	0.0	0.03	50.6	83.9	46.7	0.1	0.0	34.1	48.1	14.0	41.9	71.5	14.0
0.1	medium1	0.0	0.01	50.6	84.5	46.5	0.0	0.0	31.9	0.0	0.0	41.9	71.5	14.0
0.1	light1	0.0	0.03	50.4	83.5	45.4	0.1	0.0	33.4	56.7	16.0	38.6	77.2	16.0
0.1	light1	0.0	0.01	50.4	84.4	45.7	0.0	0.0	29.4	0.5	0.0	38.6	77.2	16.0
0.1	medium2	0.0	0.03	50.4	83.9	46.0	0.1	0.0	33.5	52.0	14.0	46.0	81.5	17.0
0.1	medium2	0.0	0.01	50.4	84.4	48.3	0.1	0.0	30.1	0.0	0.0	46.0	81.5	17.0
0.03	medium2	0.0	0.03	50.4	84.5	43.5	0.2	0.0	33.1	58.5	13.0	67.9	92.9	17.0
0.03	medium2	0.0	0.01	50.4	84.0	49.7	0.6	0.0	33.5	50.9	12.0	67.9	92.9	17.0
0.03	strong1	0.0	0.03	50.3	84.2	44.4	0.1	0.0	31.6	62.4	14.0	68.3	95.4	17.0
0.03	strong1	0.0	0.01	50.3	84.0	48.5	0.1	0.0	29.6	36.1	11.0	68.3	95.4	17.0
0.03	strong2	0.0	0.03	50.2	84.1	45.3	0.0	0.0	33.2	43.6	11.0	74.6	95.5	17.0
0.03	strong2	0.0	0.01	50.2	83.7	51.4	0.0	0.0	31.6	25.4	7.0	74.6	95.5	17.0
0.1	strong1	0.0	0.03	49.9	83.5	45.9	0.0	0.0	34.3	38.9	9.0	41.7	72.9	14.0
0.1	strong1	0.0	0.01	49.9	84.0	48.5	0.2	0.0	30.3	0.0	0.0	41.7	72.9	14.0
0.03	medium1	0.0	0.03	49.7	84.0	43.6	0.1	0.0	36.3	53.8	14.0	58.9	94.5	17.0
0.03	medium1	0.0	0.01	49.7	84.3	48.9	0.0	0.0	36.3	47.5	13.0	58.9	94.5	17.0
0.1	strong2	0.0	0.03	49.6	83.6	47.2	0.1	0.0	30.9	38.9	10.0	47.4	82.6	17.0
0.1	strong2	0.0	0.01	49.6	83.9	48.8	0.2	0.0	28.5	0.0	0.0	47.4	82.6	17.0
0.03	light1	0.1	0.03	49.5	83.7	44.4	0.5	0.0	34.1	69.9	15.0	77.7	97.6	17.0
0.03	light1	0.1	0.01	49.5	83.7	46.2	0.1	0.0	32.9	24.6	6.0	77.7	97.6	17.0
0.1	light1	0.1	0.03	49.3	83.6	44.1	0.2	0.0	34.2	51.1	12.0	53.7	72.2	17.0
0.1	light1	0.1	0.01	49.3	84.1	45.6	0.1	0.0	30.6	0.0	0.0	53.7	72.2	17.0
0.03	medium1	0.1	0.03	49.2	83.9	45.5	0.1	0.0	32.4	39.1	9.0	73.6	91.5	17.0
0.03	medium1	0.1	0.01	49.2	83.4	50.4	0.1	0.0	31.4	0.6	0.0	73.6	91.5	17.0
0.1	light0	0.1	0.03	49.0	82.7	50.4	0.0	0.0	44.7	75.7	17.0	51.4	59.2	16.0
0.1	light0	0.1	0.01	49.0	83.3	49.3	0.0	0.0	34.7	1.0	1.0	51.4	59.2	16.0
0.03	light0	0.1	0.03	48.9	82.9	50.9	0.0	0.0	45.3	75.4	16.0	64.7	96.1	17.0
0.03	light0	0.1	0.01	48.9	83.2	52.1	0.0	0.0	41.8	53.2	12.0	64.7	96.1	17.0
0.03	light1	0.0	0.03	48.5	83.7	43.3	1.2	0.0	36.0	67.4	15.0	57.1	92.2	17.0
0.03	light1	0.0	0.01	48.5	84.2	45.3	0.2	0.0	34.5	57.0	13.0	57.1	92.2	17.0
0.03	medium2	0.1	0.03	48.5	83.8	46.4	0.1	0.0	33.1	40.1	10.0	84.4	96.6	17.0
0.03	medium2	0.1	0.01	48.5	83.2	49.5	0.1	0.0	30.9	0.6	0.0	84.4	96.6	17.0
0.1	medium1	0.1	0.03	48.2	83.5	48.2	0.2	0.0	36.0	43.2	10.0	56.9	76.8	15.0
0.1	medium1	0.1	0.01	48.2	83.6	48.9	0.3	0.0	31.6	0.0	0.0	56.9	76.8	15.0
0.03	strong1	0.1	0.03	48.1	83.3	47.9	0.1	0.0	38.2	71.7	15.0	86.7	98.5	17.0
0.03	strong1	0.1	0.01	48.1	82.8	50.8	0.4	0.0	34.9	18.6	4.0	86.7	98.5	17.0
0.1	medium2	0.1	0.03	47.7	83.3	47.1	0.1	0.0	35.0	27.5	6.0	62.8	88.9	17.0
0.1	medium2	0.1	0.01	47.7	83.3	49.9	0.3	0.0	31.8	0.0	0.0	62.8	88.9	17.0
0.03	strong2	0.1	0.03	47.6	82.9	51.1	0.4	0.0	39.3	32.1	9.0	86.8	99.9	17.0
0.03	strong2	0.1	0.01	47.6	82.2	56.8	0.3	0.0	37.6	0.0	0.0	86.8	99.9	17.0
0.1	light0	0.0	0.03	47.6	81.9	54.4	0.5	0.0	45.9	83.1	17.0	50.7	78.7	16.0
0.1	light0	0.0	0.01	47.6	82.9	49.5	0.1	0.0	40.9	0.0	0.0	50.7	78.7	16.0
0.1	none	0.1	0.03	47.3	81.5	51.3	0.2	0.0	39.2	56.3	13.0	42.6	46.3	11.0
0.1	none	0.1	0.01	47.3	82.5	47.7	0.3	0.0	33.8	0.0	0.0	42.6	46.3	11.0
0.1	strong1	0.1	0.03	47.1	82.3	52.6	0.1	0.0	41.0	44.6	13.0	64.2	81.9	15.0
0.1	strong1	0.1	0.01	47.1	82.4	56.1	0.2	0.0	36.0	0.2	0.0	64.2	81.9	15.0
0.1	strong2	0.1	0.03	46.9	82.5	51.5	0.1	0.0	41.3	47.1	15.0	67.3	92.5	17.0
0.1	strong2	0.1	0.01	46.9	82.4	54.6	0.2	0.0	36.0	0.0	0.0	67.3	92.5	17.0
0.03	light0	0.0	0.03	45.0	81.5	53.5	0.3	0.0	48.7	77.7	17.0	55.8	93.0	17.0
0.03	light0	0.0	0.01	45.0	82.5	53.4	0.3	0.0	46.9	1.0	0.0	55.8	93.0	17.0
0.1	none	0.0	0.03	42.8	78.6	58.6	1.5	0.0	47.2	59.9	14.0	49.8	57.7	13.0
0.1	none	0.0	0.01	42.8	80.3	53.3	2.3	1.0	41.9	0.1	0.0	49.8	57.7	13.0
0.03	none	0.0	0.03	40.1	78.1	61.7	1.2	0.0	52.3	57.4	13.0	55.8	72.0	14.0
0.03	none	0.0	0.01	40.1	79.4	60.0	1.2	0.0	51.1	0.0	0.0	55.8	72.0	14.0

Table 4: Detailed hyperparameters and AUC values for the ViT-B/16 models with ImageNet-21k pretraining. In the *Mahalanobis (pre)* columns we evaluate the pretraining checkpoint (i.e. *before* finetuning) with Mahalanobis distance. As suggested by Bitterwolf et al. (2023), we also report the number of failed unit tests (with FPR values ≥ 10%).

						MaxLogit			**Mahalanobis**			**Maha (pre)**		
						NINCO	unit tests		NINCO	unit tests		NINCO	unit tests	
wd	aug	do	adapt-lr	pre-acc	ft-acc	AUC [%]	AUC [%]	n failed	AUC [%]	AUC [%]	n failed	AUC [%]	AUC [%]	n failed
0.1	medium1	0.0	0.03	50.6	83.9	90.4	99.9	0.0	93.4	80.7	14.0	91.3	50.0	14.0
0.1	medium1	0.0	0.01	50.6	84.5	90.6	99.9	0.0	94.0	99.9	0.0	91.3	50.0	14.0
0.1	light1	0.0	0.03	50.4	83.5	90.5	99.9	0.0	93.6	77.4	16.0	92.4	53.0	16.0
0.1	light1	0.0	0.01	50.4	84.4	91.0	99.9	0.0	94.3	99.9	0.0	92.4	53.0	16.0
0.1	medium2	0.0	0.03	50.4	83.9	90.4	99.9	0.0	93.4	86.3	14.0	89.7	46.1	17.0
0.1	medium2	0.0	0.01	50.4	84.4	90.3	99.9	0.0	94.1	99.9	0.0	89.7	46.1	17.0
0.03	medium2	0.0	0.03	50.4	84.5	91.1	99.8	0.0	93.5	85.1	13.0	79.4	33.2	17.0
0.03	medium2	0.0	0.01	50.4	84.0	89.4	99.8	0.0	93.4	89.8	12.0	79.4	33.2	17.0
0.03	strong1	0.0	0.03	50.3	84.2	91.0	99.9	0.0	93.8	83.4	14.0	81.3	36.6	17.0
0.03	strong1	0.0	0.01	50.3	84.0	89.9	99.9	0.0	94.1	90.6	11.0	81.3	36.6	17.0
0.03	strong2	0.0	0.03	50.2	84.1	91.1	99.9	0.0	93.6	89.4	11.0	79.5	37.9	17.0
0.03	strong2	0.0	0.01	50.2	83.7	89.3	99.9	0.0	94.0	93.8	7.0	79.5	37.9	17.0
0.1	strong1	0.0	0.03	49.9	83.5	90.8	99.9	0.0	93.1	86.3	9.0	91.4	59.5	14.0
0.1	strong1	0.0	0.01	49.9	84.0	90.1	99.8	0.0	94.0	99.9	0.0	91.4	59.5	14.0
0.03	medium1	0.0	0.03	49.7	84.0	90.9	99.9	0.0	92.6	75.2	14.0	81.4	23.8	17.0
0.03	medium1	0.0	0.01	49.7	84.3	90.2	99.8	0.0	92.6	85.4	13.0	81.4	23.8	17.0
0.1	strong2	0.0	0.03	49.6	83.6	90.1	99.8	0.0	93.8	92.0	10.0	90.1	47.7	17.0
0.1	strong2	0.0	0.01	49.6	83.9	90.0	99.7	0.0	94.6	99.8	0.0	90.1	47.7	17.0
0.03	light1	0.1	0.03	49.5	83.7	91.1	99.7	0.0	93.2	76.4	15.0	77.9	45.1	17.0
0.03	light1	0.1	0.01	49.5	83.7	90.9	99.8	0.0	93.6	94.2	6.0	77.9	45.1	17.0
0.1	light1	0.1	0.03	49.3	83.6	91.3	99.9	0.0	93.4	92.0	12.0	89.7	71.0	17.0
0.1	light1	0.1	0.01	49.3	84.1	91.1	99.9	0.0	93.9	99.9	0.0	89.7	71.0	17.0
0.03	medium1	0.1	0.03	49.2	83.9	90.9	99.9	0.0	93.7	94.8	9.0	80.5	44.6	17.0
0.03	medium1	0.1	0.01	49.2	83.4	89.9	99.9	0.0	93.7	99.6	0.0	80.5	44.6	17.0
0.1	light0	0.1	0.03	49.0	82.7	89.7	99.9	0.0	90.8	76.1	17.0	89.3	92.3	16.0
0.1	light0	0.1	0.01	49.0	83.3	90.5	99.8	0.0	93.1	99.3	1.0	89.3	92.3	16.0
0.03	light0	0.1	0.03	48.9	82.9	89.9	99.9	0.0	91.2	69.6	16.0	82.8	41.9	17.0
0.03	light0	0.1	0.01	48.9	83.2	89.9	99.9	0.0	92.2	88.6	12.0	82.8	41.9	17.0
0.03	light1	0.0	0.03	48.5	83.7	91.5	99.7	0.0	93.0	65.6	15.0	83.1	23.0	17.0
0.03	light1	0.0	0.01	48.5	84.2	90.9	99.8	0.0	93.5	81.4	13.0	83.1	23.0	17.0
0.03	medium2	0.1	0.03	48.5	83.8	90.5	99.9	0.0	93.6	93.5	10.0	77.2	44.2	17.0
0.03	medium2	0.1	0.01	48.5	83.2	89.8	99.8	0.0	94.2	99.2	0.0	77.2	44.2	17.0
0.1	medium1	0.1	0.03	48.2	83.5	90.5	99.9	0.0	93.1	93.4	10.0	87.1	61.7	15.0
0.1	medium1	0.1	0.01	48.2	83.6	90.4	99.8	0.0	93.8	99.9	0.0	87.1	61.7	15.0
0.03	strong1	0.1	0.03	48.1	83.3	90.4	99.9	0.0	92.4	85.3	15.0	77.1	45.9	17.0
0.03	strong1	0.1	0.01	48.1	82.8	89.8	99.7	0.0	93.1	97.7	4.0	77.1	45.9	17.0
0.1	medium2	0.1	0.03	47.7	83.3	90.3	99.9	0.0	92.7	96.1	6.0	84.6	48.9	17.0
0.1	medium2	0.1	0.01	47.7	83.3	89.9	99.8	0.0	93.4	100.0	0.0	84.6	48.9	17.0
0.03	strong2	0.1	0.03	47.6	82.9	89.7	99.9	0.0	92.4	93.3	9.0	74.1	40.4	17.0
0.03	strong2	0.1	0.01	47.6	82.2	87.9	99.8	0.0	92.5	99.6	0.0	74.1	40.4	17.0
0.1	light0	0.0	0.03	47.6	81.9	89.1	99.8	0.0	91.0	50.5	17.0	89.7	70.4	16.0
0.1	light0	0.0	0.01	47.6	82.9	90.3	99.8	0.0	92.3	100.0	0.0	89.7	70.4	16.0
0.1	none	0.1	0.03	47.3	81.5	89.5	99.8	0.0	92.1	77.0	13.0	91.3	88.8	11.0
0.1	none	0.1	0.01	47.3	82.5	90.3	99.8	0.0	93.5	99.9	0.0	91.3	88.8	11.0
0.1	strong1	0.1	0.03	47.1	82.3	89.4	99.8	0.0	91.9	92.8	13.0	84.5	53.8	15.0
0.1	strong1	0.1	0.01	47.1	82.4	88.8	99.8	0.0	92.9	99.4	0.0	84.5	53.8	15.0
0.1	strong2	0.1	0.03	46.9	82.5	89.6	99.8	0.0	92.1	94.8	15.0	82.9	46.1	17.0
0.1	strong2	0.1	0.01	46.9	82.4	89.3	99.8	0.0	93.1	99.6	0.0	82.9	46.1	17.0
0.03	light0	0.0	0.03	45.0	81.5	88.5	99.8	0.0	89.4	46.6	17.0	84.0	17.2	17.0
0.03	light0	0.0	0.01	45.0	82.5	89.0	99.7	0.0	90.3	99.8	0.0	84.0	17.2	17.0
0.1	none	0.0	0.03	42.8	78.6	87.5	99.4	0.0	89.3	55.1	14.0	88.5	57.9	13.0
0.1	none	0.0	0.01	42.8	80.3	89.0	99.4	1.0	91.7	99.9	0.0	88.5	57.9	13.0
0.03	none	0.0	0.03	40.1	78.1	87.1	99.6	0.0	88.1	59.0	13.0	84.2	34.2	14.0
0.03	none	0.0	0.01	40.1	79.4	87.8	99.6	0.0	89.3	99.9	0.0	84.2	34.2	14.0

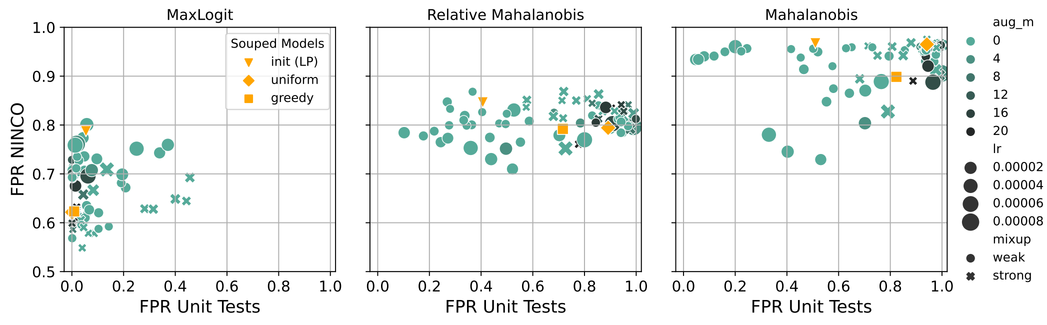

Figure 5: Clip Models. Souped models are shown in orange.

B.4 Evaluating more methods

In Figures 6, 7and 8 we plot results for more methods than the ones investigated in the main paper and sketch some of the high-level trends.

Pretraining on ImageNet-21k and finetuning on ImageNet-1k . The logit-based methods (first row in Figure 6) all perform relatively well on the unit tests, with MaxLogit and Energy solving the task consistently for all models (FPR values of basically 0%). For MSP and KL-Matching, the FPR values are not exactly 0, but below 10% with few exceptions. The results on NINCO vary and do not show conclusive trends with respect to training parameters.

For feature-based methods (second row in Figure 6), the strongest method on NINCO is ViM, where a subset of models achieves FPR values below 40%, and of those several simultaneously yield close to 0% FPR on the unit tests. Interestingly, augmentations in combination with dropout now seem to play a more important role than learning rate and weight decay. This is likely due to the fact that ViM uses *both* feature and logit information. Still, many models yield FPR values larger than 20% FPR. For Energy+React, consistently good results are achieved on the unit tests, and several models also perform strongly on NINCO (e.g. those trained with augmentation type *light1* and no dropout all achieve FPR values of 34-37%).

Training exclusively on ImageNet-1k. For this pretraining setup (Figure 7), the models do not generally achieve 0% FPR on the unit tests for any of the displayed methods. In contrast to ImageNet-21k pretraining, feature-based methods loose their edge over logit-based methods on NINCO . For almost all setups, stronger augmentations improve OOD detection.

Pretraining on CLIP and finetuning on ImageNet-1k. For CLIP-pretrained models (Figure 8) out of the feature-based methods only Energy+React works reasonably well on NINCO and the unit tests. For this setup, the souped models (orange in plots) are amongst the better detectors, but they fail completely for e.g. ViM, knn and cosine. For Energy+React and MaxLogit, the overall best detectors, smaller learning rate leads to improved performnace on NINCO.

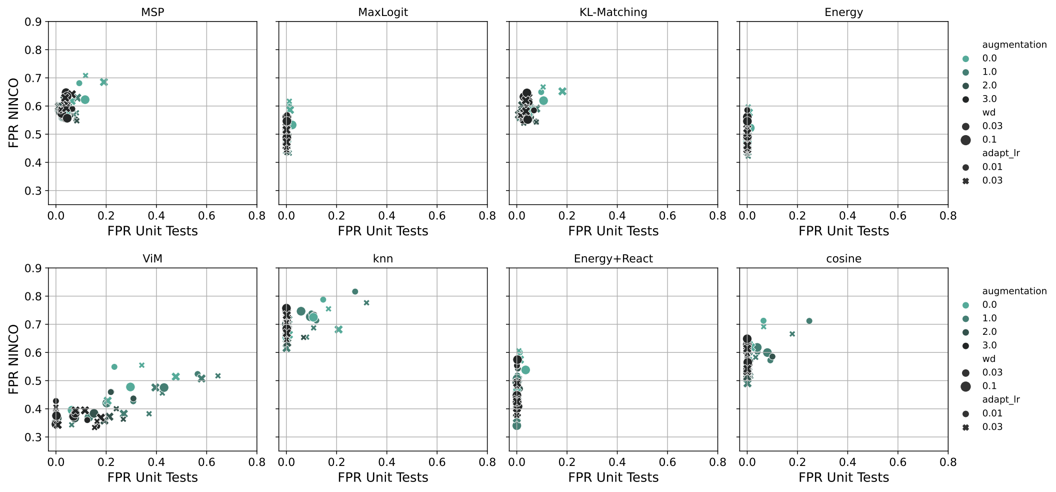

Figure 6: ImageNet-1k models pretrained on ImageNet-21k with more methods (logit-based on the top, feature-based on the bottom).

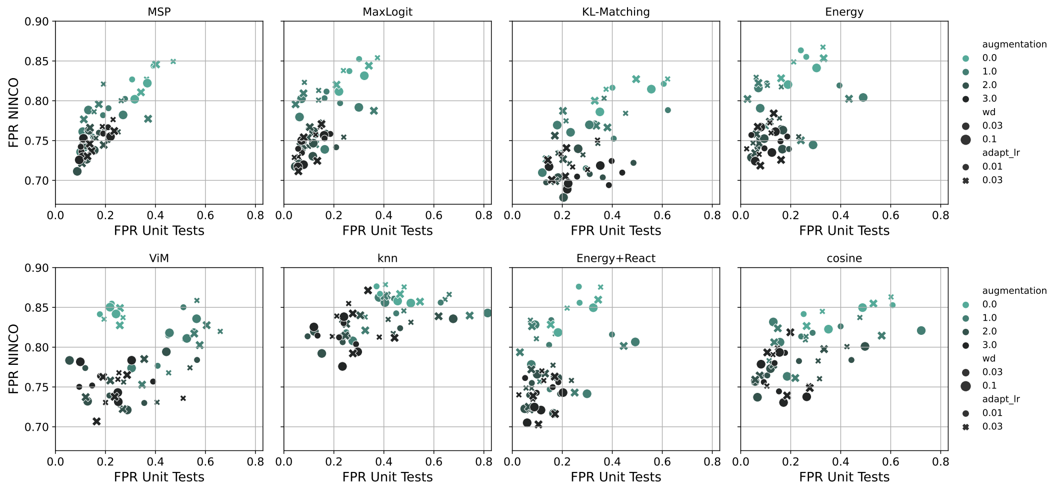

Figure 7: ImageNet-1k models without pretraining, evaluated with more methods (logit-based on the top, feature-based on the bottom).

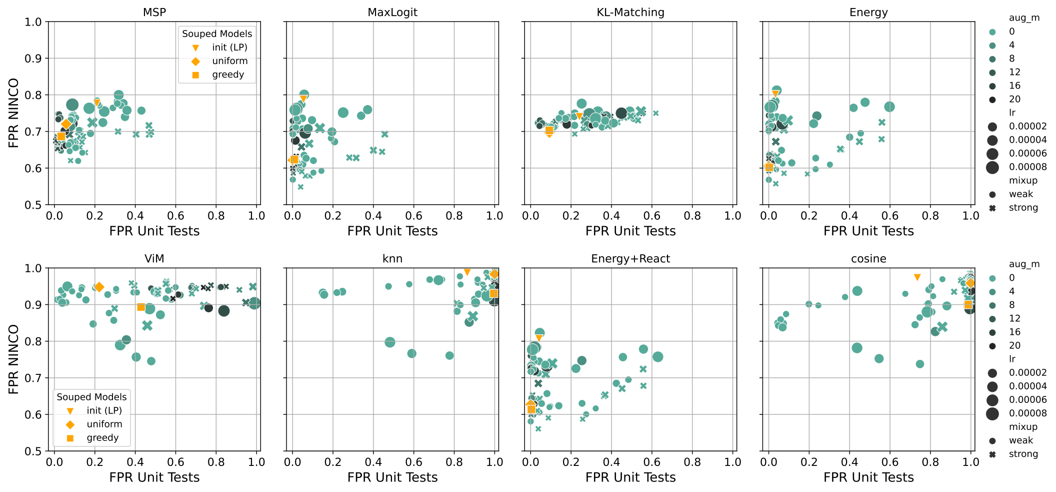

Figure 8: ImageNet-1k models with CLIP-pretraining evaluated with more methods (logit-based on the top, feature-based on the bottom, souped models in orange).

C Methods

Here we give an overview over the evaluated OOD detection methods. For clarity, we denote vectors in bold and lowercase letters and matrices in bold an uppercase letters. We write neural networks as functions n, which are parametrized by weights θ, take an input sample $\mathbf{x}$ and produce an output vector $\mathbf{o}$ of size C, where C is typically the number of classes in a classification task (1000 in the case of IN-1K). We refer to $\mathbf{o}$ as the logits of $\mathbf{x}$, which can be transformed to a probability vector $\mathbf{p}$ (also of size C) via the softmax function: $p_i = \exp(o_i) / \sum_c \exp(o_c)$. The network n can be decomposed into a feature extractor h and the networks last layer g:

$$\mathbf{o} = n(\mathbf{x}) = g(h(\mathbf{x})),$$

where g is a fully connected, linear layer, i.e. $g(\mathbf{h}) = \mathbf{W}^T\mathbf{h} + \mathbf{b}$ with weight $\mathbf{W}$ and bias $\mathbf{b}$. We refer to $\mathbf{h} = h(\mathbf{x})$ as the *features* or the *embeddings* of $\mathbf{x}$ w.r.t. the network n. For each sample $\mathbf{x}$, a method returns an OOD-score $s = f(\mathbf{x})$, a scalar value which is supposed to be larger for ID data and smaller for OOD data. Methods accessing $h(\mathbf{x})$ directly in order to compute the OOD-score are referred to as feature-based methods, in contrast to methods that derive their OOD-score from the logits $\mathbf{o}$ (even though obviously the logits implicitly also depend on these features). In the following, we will describe how each method computes the score s for a test input $\mathbf{x}$.

MSP Hendrycks and Gimpel (2017): The most popular OOD-detection baseline uses the confidence, i.e. the max softmax probability of a models probability output vector:

$$s = \max_c(p_c)$$

Max-Logit Hendrycks et al. (2022): Similar to MSP, Max-Logit returns the largest entry of the logit-vector $\mathbf{o}$, i.e.

$$s = \max_c(o_c)$$

Energy Liu et al. (2020): The Energy based OOD detection method uses the denominator of the softmax-function as OOD-score:

$$s = \log \sum_c^C \exp(o_c)$$

KL-Matching Hendrycks et al. (2022): KL-Matching computes a mean probability vector $\mathbf{d}_c$ for each of the C classes. For a test input, the KL-distances of all $\mathbf{d}_c$ vectors to its probability vector $\mathbf{p}$ are computed, and the OOD-score is the negative of the smallest of those distances:

$$s = -\min_c \mathrm{KL}[\mathbf{p}||\mathbf{d}_c]$$

In the original paper by Hendrycks et al. (2022), the average for $\mathbf{d}_c$ is computed over an additional validation set. Since none of the other methods leverages extra data and we are interested in fair comparison, we deploy KL-Matching like in Wang et al. (2022); Yang et al. (2022), where the average is computed over the train set.

KNN Sun et al. (2022): KNN is a non-parametric method that computes distances in the feature-space. Specifically, the feature vector of a test input is normalized to $\mathbf{z} = \mathbf{h}/||\mathbf{h}||_2$ and the pairwise distances $r_i(\mathbf{z}) = ||\mathbf{z} - \mathbf{z}_i||_2$ to the normalized features $\mathcal{Z} = \{\mathbf{z}_1, ..., \mathbf{z}_N\}$ of all samples of the training set are computed. The distances $r_i(\mathbf{z})$ are then sorted according to their magnitude and the K^{th} smallest distance, denoted $r^K(\mathbf{z})$ is used as negative OOD-score:

$$s = -r^K(\mathbf{z})$$

Like suggested in Sun et al. (2022), we use $K = 1000$.

Mahalanobis distance Lee et al. (2018): This popular method fits a class-conditional Gaussian with shared covariance matrix to the train set, i.e. computes

$$\hat{\mu}_c = \frac{1}{N_c} \sum_{i:y_i=c} \mathbf{h}_i, \qquad \hat{\Sigma} = \frac{1}{N} \sum_c \sum_{i:y_i=c} (\mathbf{h}_i - \hat{\mu}_c)(\mathbf{h}_i - \hat{\mu}_c)^T$$

where N_c is the number of train samples in class c and N is the total number of train samples. The OOD-score of a test sample is then the Mahalanobis distance induced by $\hat{\Sigma}$ between its feature $\mathbf{h}$ and the closest class mean:

$$s = -\min_c (\mathbf{h} - \hat{\mu}_c)\hat{\Sigma}^{-1}(\mathbf{h} - \hat{\mu}_c)^T$$

Relative Mahalanobis distance Ren et al. (2021): A modification of the Mahalanobis distance method, thought to improve near-OOD detection, is to additionally fit a global Gaussian distribution to the train set without taking class-information into account:

$$\hat{\mu}_{\text{global}} = \frac{1}{N}\sum_i \mathbf{h}_i, \qquad \hat{\Sigma}_{\text{global}} = \frac{1}{N}\sum_i (\mathbf{h}_i - \hat{\mu}_{\text{global}})(\mathbf{h}_i - \hat{\mu}_{\text{global}})^T$$

The OOD-score is then defined as the difference between the original Mahalanobis distance and the Mahalanobis distance w.r.t. the global Gaussian distribution:

$$s = -\min_c \left((\mathbf{h} - \hat{\mu}_c)\hat{\Sigma}^{-1}(\mathbf{h} - \hat{\mu}_c)^T - (\mathbf{h} - \hat{\mu}_{\text{global}})\hat{\Sigma}^{-1}_{\text{global}}(\mathbf{h} - \hat{\mu}_{\text{global}})^T \right)$$

ReAct Sun et al. (2021): The authors propose to perform a truncation of the feature vector, $\bar{\mathbf{h}} = \min(\mathbf{h}, r)$, where the min operation is to be understood element-wise and r is the truncation threshold. The truncated features can then be converted to so-called rectified logits via $\bar{\mathbf{o}} = g(\bar{\mathbf{h}}) = \mathbf{W}^T\bar{\mathbf{h}} + \mathbf{b}$. While the rectified logits can now be used with a variety of existing detection methods, we follow Sun et al. (2021) and use the rectified Energy as OOD-score:

$$s = \log \sum_c^C \exp(\bar{o}_c)$$

As suggested in Wang et al. (2022), we set the threshold r such that 1% of the activations from the train set would be truncated.

Virtual Logit Matching Wang et al. (2022): The idea behind ViM is that meaningful features are thought to lie in a low-dimensional manifold, called the principal space P, whereas features from OOD-samples should also lie in $P^\perp$, the space orthogonal to P. P is the D-dimensional subspace spanned by the eigenvectors with the largest D eigenvalues of the matrix $\mathbf{F}^T\mathbf{F}$, where $\mathbf{F}$ is the matrix of all train features offsetted by $\mathbf{u} = -(\mathbf{W}^\mathbf{T})^+\mathbf{b}$ (+ denotes the Moore-Penrose inverse). A sample with feature vector $\mathbf{h}$ is then also offset to $\tilde{\mathbf{h}} = \mathbf{h} - \mathbf{u}$ and can be decomposed into $\tilde{\mathbf{h}} = \tilde{\mathbf{h}}^P + \tilde{\mathbf{h}}^{P^\perp}$, and $\tilde{\mathbf{h}}^{P^\perp}$ is referred to as the *Residual* of $\mathbf{h}$. ViM leverages the Residual and converts it to a virtual logit $o_0 = \alpha||\tilde{\mathbf{h}}^{P^\perp}||_2$, where

$$\alpha = \frac{\sum_{i=1}^N \max_c o_i^c}{\sum_{i=1}^N ||\mathbf{h}_i^{P^\perp}||_2}$$

is designed to match the scale of the virtual logit to the scale of the real train logits. The virtual logit is then appended to the original logits of the test sample, i.e. to $\mathbf{o}$, and a new probability vector is computed via the softmax function. The probability corresponding to the virtual logit is then the final OOD-score:

$$s = -\frac{\exp(o_0)}{\sum_{c=1}^C \exp(o_c) + \exp(o_0)}$$

Like suggested in Wang et al. (2022), we use $D = 1000$ if the dimensionality of the feature space d is $d \geq 2048$, $D = 512$ if $2048 \geq d \geq 768$, and $D = d/2$ rounded to integers otherwise.

Cosine Tack et al. (2020); Galil et al. (2023): This method computes the maximum cosine-similarity between the features of a test-sample and embedding vectors $\tilde{\mathbf{u}}_c$ (sometimes also called concept-vector):

$$s = \max_c \tilde{\mathbf{u}}_c^T \mathbf{h} / ||\tilde{\mathbf{u}}_c^T||_2 \tag{1}$$

For zero-shot CLIP, $\tilde{\mathbf{u}}_c$ can be obtained by creating text-embeddings from the ImageNet class names. Encoding *'A photo of a ...'* yields an embedding from the corresponding class. For classifiers, we use the class-wise train means $\hat{\mu}_c$, that are also used for Mahalanobis distance.

D Definitions of OOD detection metrics

The performance of OOD detectors is commonly reported in terms of the *false positive rate at a fixed true positive rate* Q, denoted as **FPR@TPRQ**, short **FPR**. This means that the detector is interpreted as making the decision to *accept* an unknown input x *if* $S(x) \geq \tau$, for a threshold τ

that is chosen such that Q% of ID inputs are accepted, and *rejecting* the input as OOD *if* $S(x) < \tau$. The FPR@TPRQ counts the fraction of falsely accepted OOD inputs under this decision scheme. This means the *lower* the FPR@TPRQ, the *better* the OOD detection performance. In the OOD detection literature, the most commonly used value for Q is 95%, which we too use throughout this paper. As suggested in Bitterwolf et al. (2023), we compute the FPR value for each NINCO class separately, and report the average FPR across classes. For the unit-tests, the authors suggest reporting the number of failed unit tests (defined by a FPR threshold of 10%). Since this would lead to overlapping data points in our plots, we decided to show the mean FPR across the unit-test classes instead in the figures. For Table 3, we also report the number of failed unit tests.