Vision-LSTM: xLSTM as Generic Vision Backbone

Benedikt Alkin [1 2 3] **Maximilian Beck** [1 2 3] **Korbinian Pöppel** [1 2 3] **Sepp Hochreiter** [1 2 3] **Johannes Brandstetter** [1 2 3]

Abstract

Transformers are widely used as generic backbones in computer vision, despite initially introduced for natural language processing. Recently, the Long Short-Term Memory (LSTM) has been extended to a scalable and performant architecture – the xLSTM – which overcomes long-standing LSTM limitations via exponential gating and parallelizable matrix memory structure. In this report, we introduce Vision-LSTM (ViL), an adaption of the xLSTM building blocks to computer vision. ViL comprises a stack of xLSTM blocks where odd blocks process the sequence of patch tokens from top to bottom while even blocks go from bottom to top. Experiments show that ViL holds promise to be further deployed as new generic backbone for computer vision architectures.
Project page: nx-ai.github.io/vision-lstm

1. Introduction

Language modeling architectures — such as Transformers (Vaswani et al., 2017; Achiam et al., 2023; Team et al., 2023) or more recently State Space Models (Gu et al., 2021; Gupta et al., 2022) such as Mamba (Gu & Dao, 2023) — are commonly adapted to the domain of computer vision to make use of their powerful modeling capabilities. However, in natural language processing, an input sentence is typically encoded into tokens that represent words or common subwords (Bostrom & Durrett, 2020) via a discrete vocabulary. To encode images into a set of tokens, Vision Transformer (Dosovitskiy et al., 2021) (ViT) proposed to group an input image into non-overlapping patches, linearly project them into a sequence of so-called patch tokens and add positional information to these tokens. This sequence can then be processed by language modeling architectures.

The Extended Long Short-Term Memory (xLSTM) family (Beck et al., 2024) was recently introduced as a new architecture for language modeling. It demonstrates the resurgence of LSTM in the LLM era, performing favorably against the likes of Transformers and State Space Models (SSMs). Analogous to existing vision versions of Transformers or State Space Models, e.g., ViT (Dosovitskiy et al., 2021) or Vision Mamba (Zhu et al., 2024), which have produced great results in various computer vision tasks (Singh et al., 2023; Kirillov et al., 2023; Oquab et al., 2023; Peebles & Xie, 2023; Alkin et al., 2024b), we introduce Vision LSTM (ViL) – a generic computer vision backbone that uses xLSTM blocks as its core components. To adjust xLSTM (an autoregressive model) to computer vision (an often non-autoregressive domain), we employ a stack of alternating mLSTM blocks (Beck et al., 2024) where odd blocks process patches row-wise from top left to bottom right and even blocks go from bottom right to top left. This simple alternating design allows ViL to efficiently process non-sequential inputs, such as images, without introducing additional computations.

Similar to vision adaptions of SSMs (Liu et al., 2024; Zhu et al., 2024; Wang et al., 2024), ViL can exhibit linear computational and memory complexity w.r.t. sequence length which makes it appealing for tasks that benefit from high-resolution images such as medical imaging (Chen et al., 2021; Hatamizadeh et al., 2022; Valanarasu et al., 2021; Xu et al., 2024), segmentation (Kirillov et al., 2023; Cheng et al., 2022), or physics simulations (Bi et al., 2023; Nguyen et al., 2023; Bodnar et al., 2024; Alkin et al., 2024a). In contrast, ViT's computational complexity scales quadratically due to the self-attention mechanism, rendering them costly to apply to high-resolution tasks.

2. Method

Vision-LSTM (ViL) is a generic backbone for computer vision tasks, which is residually built from xLSTM blocks, as visualized in Figure 1. Following ViT (Dosovitskiy et al., 2021), ViL first splits an image into non-overlapping patches via a shared linear projection, then adds learnable positional embeddings to each patch token. At the core of ViL are alternating mLSTM blocks, which are fully parallelizable and equipped with a matrix memory combined with a covariance update rule. Odd mLSTM blocks process patch tokens from top left to bottom right while even blocks go from bottom right to top left.

[1]ELLIS Unit Linz, Institute for Machine Learning, JKU Linz, Austria [2]NXAI Lab, Linz, Austria [3]NXAI GmbH, Linz, Austria. Correspondence to: Benedikt Alkin <alkin@ml.jku.at>.

Proceedings of the 1st Workshop on Long-Context Foundation Models, Vienna, Austria. 2024.

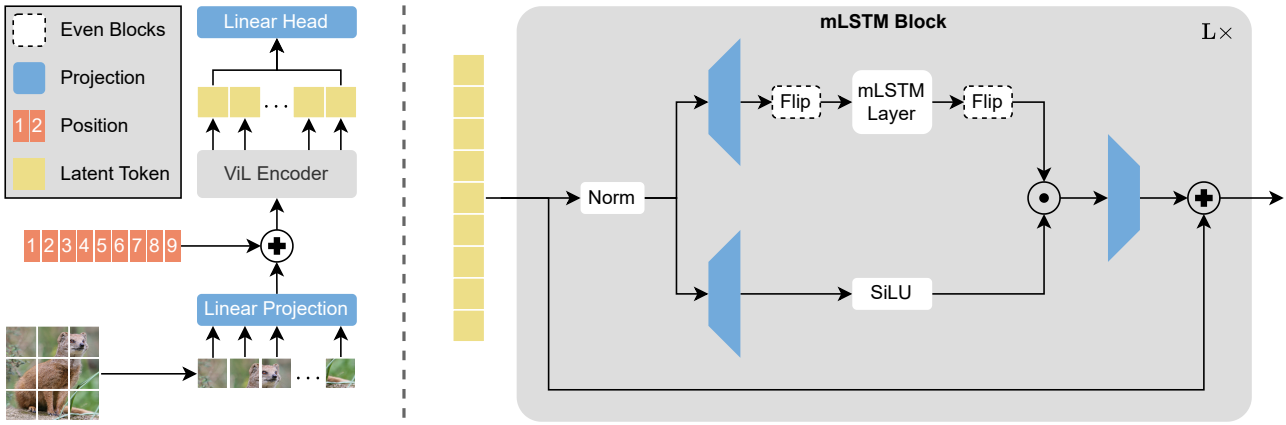

Figure 1. Schematic overview of Vision-LSTM (ViL). Following ViT (Dosovitskiy et al., 2021), an input image is split into patches and linearly projected. Then, a learnable vector is added per position to the patches, producing a sequence of patch tokens. This sequence is then processed by alternating mLSTM blocks where even blocks flip the sequence before and after the mLSTM layer. For classification, ViL uses the concatenation of the first and the last patch as input to a linear classification head.

3. Experiments

We pre-train models on ImageNet-1K (Deng et al., 2009), which contains 1.3M training images and 50K validation images where each image belongs to one of 1000 classes. ViL models are trained for 800 epochs (tiny) or 400 epochs (small, base) on 192x192 resolution with a learning rate of 1e-3 using a cosine decay schedule. Afterwards, the model is fine-tuned on 224x224 resolution for 20 epochs using a learning rate of 1e-5. Detailed hyperparameters can be found in Appendix Table 11.

As ViL is an isotropic architecture (like ViT (Dosovitskiy et al., 2021)), we compare it against other isotropic architectures and leave exploration of hierarchical ViL architectures to future work.

As ViTs are well established in the vision community, they underwent multiple optimization cycles over the years (Dosovitskiy et al., 2021; Touvron et al., 2021a; 2022a; 2021b; 2022b; He et al., 2022). Since this work is the first to apply xLSTM to computer vision, we do not expected to outperform years of hyperparameter tuning of ViTs in all cases. Even so, Figure 2 shows an overview of performance metrics in relation to total pre-training compute where ViL performs favorably against heavily optimized ViT protocols (DeiT, DeiT-III), Vim (Zhu et al., 2024) and also an isotropic version of ConvNeXt (Liu et al., 2022).

3.1. ImageNet-1K Pre-training

Table 1 relates parameter counts and FLOPS to ImageNet-1K accuracy after pre-training. ViL outperforms heavily optimized ViT protocols and other sequential backbones on the tiny and small scale. While ViL does not outperform all other models on the base scale, evaluations on downstream tasks (as shown later in Table 5 and Table 4) show that ViL-B still learns strong features. Additionally, training a ViL-B for 400 epochs takes roughly 600 A100 GPU-hours or 19 hours on 32 A100 GPUs. Therefore, the hyperparameter configuration is most likely not optimal as we were not able to extensively optimize hyperparameters for this scale. Note that training ViL is still quite expensive as there does not exist an optimized hardware implementations yet, which does exist for other models like ViT (Dao et al., 2022; Dao, 2023) or Vim (Gu & Dao, 2023; Zhu et al., 2024). Nevertheless, the efficient alternating block design of ViL makes it already faster than Vim (up to 69% as shown in Appendix A.4) while only using `torch.compile`, a generic speed optimization of the PyTorch (Paszke et al., 2019) framework.

3.2. Classification Design

In order to perform classification with a ViT, the sequence of tokens is pooled into a single token, which is then used as input to a classification head. The most common methods to do this pooling are (i) adding a learnable [CLS] token at the start of the sequence or (ii) averaging all patch tokens to produce an [AVG] token. Whether to use the [CLS] or [AVG] token is typically a hyperparameter, where both variants achieve comparable performances. On the contrary, autoregressive models often require specialized classification designs. For example, Vim (Zhu et al., 2024) requires the [CLS] token to be in the middle of the sequence, suffering heavy performance losses if other classification designs, e.g., an [AVG] token or two [CLS] tokens at start and end of the sequence, are employed. Due to its autoregressive

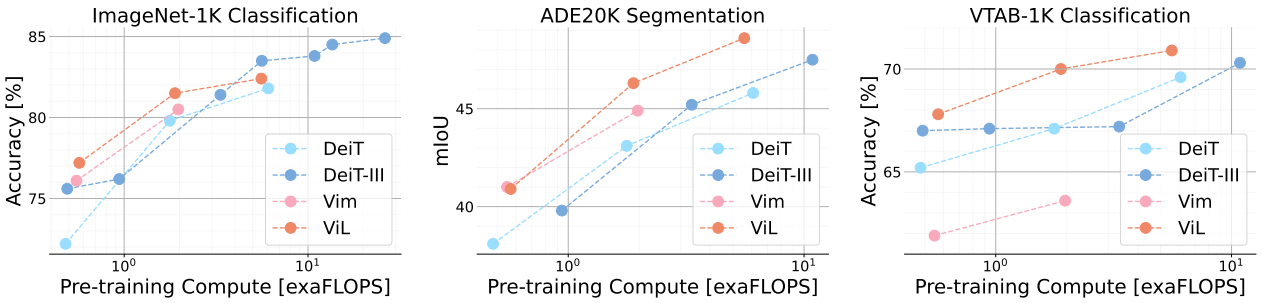

Figure 2. Performance overview of ImageNet-1K pre-trained models in relation to pre-training compute. ViL shows strong performances across classification and semantic segmentation tasks. Detailed results in Appendix Table 4 and Appendix Table 5.

Model	Epochs	#Params	FLOPS	IN-1K
DeiT-T (Touvron et al., 2021a)	300	6M	1.3G	72.2
DeiT-II-T (Touvron et al., 2022a)	400	6M	1.3G	73.5
DeiT-III-T (reimpl.)	800+20	6M	1.3G	76.2
Vim-T (Zhu et al., 2024)	300	7M	1.5G	76.1
ViL-T	800+20	6M	1.5G	**78.3**
DeiT-S (Touvron et al., 2021a)	300	22M	4.6G	79.8
DeiT-II-S (Touvron et al., 2022a)	400	22M	4.6G	80.7
DeiT-III-S (Touvron et al., 2022b)	800+20	22M	4.6G	81.4
ConvNeXt-S (Liu et al., 2022) (*iso.*)	300	22M	4.3G	79.7
Vim-S (Zhu et al., 2024)	300	26M	5.3G	80.5
ViL-S	400+20	23M	5.1G	**81.5**
DeiT-B (Touvron et al., 2021a)	300	86M	17.6G	81.8
DeiT-II-B (Touvron et al., 2022a)	400	86M	17.6G	82.7
DeiT-III-B (Touvron et al., 2022b)	800+20	86M	17.6G	**83.7**
ConvNeXt-B (Liu et al., 2022) (*iso.*)	300	87M	16.9G	82.0
ViL-B	400+5	89M	18.6G	82.4

Table 1. ImageNet-1K pre-training accuracy. All models use a patch size of 16x16 with 224x224 resolution at most. Models with "+" in their "Epochs" column pre-train on lower resolution followed by fine-tuning on 224x224 resolution for some epochs.

nature, we explore different classification designs for ViL in Table 2. [AVG] averages all patch tokens, "Middle Patch" uses the middle patch token, "Middle [CLS]" uses a [CLS] token in the middle of the sequence, "Bilateral [AVG]" uses the average of the first and last patch token. We find ViL to be relatively robust the classification design where all performances that use a single token are within 0.6%. We only add a [CLS] token if it is used for classification, hence the best design ("Bilateral Concat") does not use a [CLS] token.

Design	Dim	ACC
[AVG]	192	75.4
Middle Patch	192	**75.9**
Middle [CLS]	192	75.3
Bilateral [AVG]	192	**75.9**
Bilateral Concat	384	**76.4**

Table 2. How to pool the sequence of tokens for classification? ViL is relatively robust to the classification design. We conduct this study by training a ViL-T on ImageNet-1K for 400 epochs.

3.3. Block Design.

We study different ways to design ViL blocks in Table 3. The plain, uni-directional xLSTM block fails to reach competitive performances since the autoregressive nature of xLSTM is not suited for image classification. Traversing blocks in a bi-directional manner, i.e., introducing a second mLSTM layer in each block that traverses the sequence backwards (akin to Vim (Zhu et al., 2024)), improves performance, but also requires more parameters and FLOPS. Sharing the parameters of the forward and backward mLSTM makes the model more parameter efficient, but still requires more compute and overloads these parameters which leads to performance drops. Using alternating blocks enhances performance while remaining compute and parameter efficient. We also explore quad-directional designs (similar to (Liu et al., 2024)), which refers to traversing the sequence row-wise (in both directions) and additionally column-wise (in both directions). Bi-directional traverses the sequence only row-wise (in both directions). Figure 3 visualizes the different traversal paths.

Due to the increased cost of bi-directional and quad-

Direction	Alternating	Param. Sharing	ACC	#Param	FLOPS	Runtime
Uni-dir. (xLSTM)	✗	✗	72.8	6.1M	1.0x	1.0x
Bi-dir.	✗	✗	**74.8**	6.5M	1.4x	1.9x
Bi-dir.	✗	✓	73.3	6.1M	1.4x	1.9x
Bi-dir. (ViL)	✓	✗	**74.8**	6.1M	1.0x	1.0x
Quad-dir.	✗	✗	**76.3**	7.3M	2.3x	7.5x
Quad-dir.	✗	✓	72.8	6.1M	2.3x	7.5x
Quad-dir.	✓	✗	75.6	6.1M	1.0x	1.9x

Table 3. Traversing the sequence of patch tokens in multiple directions yields better performance than the plain, uni-directional xLSTM block. However, using multiple directions per block also requires more compute and parameters. Sharing parameters between directions matches #Params but overloads them, leading to performance drops. Using multi-directional designs but alternating between them in consecutive blocks, matches #Params and FLOPS while achieving good performances. While quad-directional designs yields the best performances, they comes with drastically increased runtime. As our current quad-directional implementation is not compatible with `torch.compile`, it suffers from increased runtime even when alternating between directions. Since quad-directional designs are compute intensive, this study was conducted by training a ViL-T on an ImageNet-1K subset of 100 classes using resolution 128. "Param. Sharing" shares the parameters of the two (bidirectional) or four (quaddirectional) mLSTM layers in each block.

directional blocks, this study was conducted in a heavily reduced setting. We train on a subset of ImageNet-1K that contains only samples from 100 classes for 400 epochs at 128x128 resolution. This was particularly necessary because our quad-directional implementation is not compatible with `torch.compile` (a generic speed optimization method from PyTorch (Paszke et al., 2019)), which leads to longer runtimes, as indicated last column of Table 3. Due to this technical limitation, we choose the alternating bi-directional block as our core design.

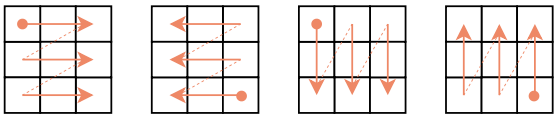

Figure 3. Visualization of traversal paths. Bi-directional uses the first two traversal paths. Quad-directional uses all of the above. Squares represent individual patch tokens.

4. Conclusion

Motivated by the success of xLSTM in language modeling, we introduced ViL, an adaption of the xLSTM architecture to vision tasks. ViL processes a sequence of patch tokens in alternating fashion. Odd blocks process image patches row-wise from top left to bottom right and even blocks go from bottom right to top left. Our new architecture outperforms SSM-based vision architectures and also optimized ViT models on ImageNet-1K classification, VTAB-1K transfer classification and ADE20K semantic segmentation. Remarkably, ViL is able to outperform ViT training pipelines, which are the result of years of hyperparameter tuning and transformer improvements.

In the future, we see potential in applying ViL when high-resolution images are needed for optimal performance, such as semantic segmentation or medical imaging. In these settings, transformers suffer from high computational costs due to the quadratic complexity of self-attention, where ViL can use a chunked form to trade-off between its parallel form (quadratic complexity) and recurrent form (linear complexity). Additionally, improving pre-training schemes (e.g. via self-supervised learning), exploring better hyperparameter settings or investigating hierarchical architectures are promising future directions that could improve ViL even further.

Acknowledgments

We acknowledge EuroHPC Joint Undertaking for awarding us access to Karolina at IT4Innovations, Czech Republic, MeluXina at LuxProvide, Luxembourg, Leonardo at CINECA, Italy and LUMI at CSC, Finland.

The ELLIS Unit Linz, the LIT AI Lab, the Institute for Machine Learning, are supported by the Federal State Upper Austria. We thank the projects Medical Cognitive Computing Center (MC3), INCONTROL-RL (FFG-881064), PRIMAL (FFG-873979), S3AI (FFG-872172), DL for GranularFlow (FFG-871302), EPILEPSIA (FFG-892171), AIRI FG 9-N (FWF-36284, FWF-36235), AI4GreenHeatingGrids (FFG- 899943), INTEGRATE (FFG-892418), ELISE (H2020-ICT-2019-3 ID: 951847), Stars4Waters (HORIZON-CL6-2021-CLIMATE-01-01). We thank Audi.JKU Deep Learning Center, TGW LOGISTICS GROUP GMBH, Silicon Austria Labs (SAL), FILL Gesellschaft mbH, Anyline GmbH, Google, ZF Friedrichshafen AG, Robert Bosch GmbH, UCB Biopharma SRL, Merck Healthcare KGaA, Verbund AG, GLS (Univ. Waterloo), Software Competence Center Hagenberg GmbH, Borealis AG, TÜV Austria, Frauscher Sensonic, TRUMPF and the NVIDIA Corporation.

References

Achiam, J., Adler, S., Agarwal, S., Ahmad, L., Akkaya, I., Aleman, F. L., Almeida, D., Altenschmidt, J., Altman, S., Anadkat, S., et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Alkin, B., Fürst, A., Schmid, S., Gruber, L., Holzleitner, M., and Brandstetter, J. Universal physics transformers. *arXiv preprint arXiv:2402.12365*, 2024a.

Alkin, B., Miklautz, L., Hochreiter, S., and Brandstetter, J. Mim-refiner: A contrastive learning boost from intermediate pre-trained representations. *arXiv preprint arXiv:2402.10093*, 2024b.

Ba, L. J., Kiros, J. R., and Hinton, G. E. Layer normalization. *CoRR*, abs/1607.06450, 2016. URL http://arxiv.org/abs/1607.06450.

Beck, M., Pöppel, K., Spanring, M., Auer, A., Prudnikova, O., Kopp, M., Klambauer, G., Brandstetter, J., and Hochreiter, S. xlstm: Extended long short-term memory, 2024.

Bi, K., Xie, L., Zhang, H., Chen, X., Gu, X., and Tian, Q. Accurate medium-range global weather forecasting with 3d neural networks. *Nature*, 619(7970):533–538, 2023.

Bodnar, C., Bruinsma, W. P., Lucic, A., Stanley, M., Brandstetter, J., Garvan, P., Riechert, M., Weyn, J., Dong, H., Vaughan, A., et al. Aurora: A foundation model of the atmosphere. *arXiv preprint arXiv:2405.13063*, 2024.

Bostrom, K. and Durrett, G. Byte pair encoding is suboptimal for language model pretraining. *arXiv preprint arXiv:2004.03720*, 2020.

Chen, J., Lu, Y., Yu, Q., Luo, X., Adeli, E., Wang, Y., Lu, L., Yuille, A. L., and Zhou, Y. Transunet: Transformers make strong encoders for medical image segmentation. *CoRR*, abs/2102.04306, 2021.

Cheng, B., Misra, I., Schwing, A. G., Kirillov, A., and Girdhar, R. Masked-attention mask transformer for universal image segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, LA, USA, June 18-24, 2022*, pp. 1280–1289. IEEE, 2022.

Cheng, G., Han, J., and Lu, X. Remote sensing image scene classification: Benchmark and state of the art. *Proc. IEEE*, 105(10):1865–1883, 2017.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., Schuh, P., Shi, K., Tsvyashchenko, S., Maynez, J., Rao, A., Barnes, P., Tay, Y., Shazeer, N., Prabhakaran, V., Reif, E., Du, N., Hutchinson, B., Pope, R., Bradbury, J., Austin, J., Isard, M., Gur-Ari, G., Yin, P., Duke, T., Levskaya, A., Ghemawat, S., Dev, S., Michalewski, H., Garcia, X., Misra, V., Robinson, K., Fedus, L., Zhou, D., Ippolito, D., Luan, D., Lim, H., Zoph, B., Spiridonov, A., Sepassi, R., Dohan, D., Agrawal, S., Omernick, M., Dai, A. M., Pillai, T. S., Pellat, M., Lewkowycz, A., Moreira, E., Child, R., Polozov, O., Lee, K., Zhou, Z., Wang, X., Saeta, B., Diaz, M., Firat, O., Catasta, M., Wei, J., Meier-Hellstern, K., Eck, D., Dean, J., Petrov, S., and Fiedel, N. Palm: Scaling language modeling with pathways. *J. Mach. Learn. Res.*, 24:240:1–240:113, 2023.

Clark, K., Luong, M., Le, Q. V., and Manning, C. D. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

Dao, T. Flashattention-2: Faster attention with better parallelism and work partitioning. *CoRR*, abs/2307.08691, 2023.

Dao, T., Fu, D. Y., Ermon, S., Rudra, A., and Ré, C. Flashattention: Fast and memory-efficient exact attention with io-awareness. In *NeurIPS*, 2022.

Dehghani, M., Djolonga, J., Mustafa, B., Padlewski, P., Heek, J., Gilmer, J., Steiner, A. P., Caron, M., Geirhos, R., Alabdulmohsin, I., Jenatton, R., Beyer, L., Tschannen, M., Arnab, A., Wang, X., Ruiz, C. R., Minderer, M., Puigcerver, J., Evci, U., Kumar, M., van Steenkiste, S., Elsayed, G. F., Mahendran, A., Yu, F., Oliver, A., Huot, F., Bastings, J., Collier, M., Gritsenko, A. A., Birodkar, V., Vasconcelos, C. N., Tay, Y., Mensink, T., Kolesnikov, A., Pavetic, F., Tran, D., Kipf, T., Lucic, M., Zhai, X., Keysers, D., Harmsen, J. J., and Houlsby, N. Scaling vision transformers to 22 billion parameters. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J. (eds.), *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pp. 7480–7512. PMLR, 2023.

Deng, J., Dong, W., Socher, R., Li, L., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR 2009), 20-25 June 2009, Miami, Florida, USA*, pp. 248–255. IEEE Computer Society, 2009.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. An image is worth 16x16 words: Transformers for image

recognition at scale. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

Fei-Fei, L., Fergus, R., and Perona, P. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, 28(4):594–611, 2006.

Geiger, A., Lenz, P., Stiller, C., and Urtasun, R. Vision meets robotics: The KITTI dataset. *Int. J. Robotics Res.*, 32(11):1231–1237, 2013.

Goyal, P., Dollár, P., Girshick, R. B., Noordhuis, P., Wesolowski, L., Kyrola, A., Tulloch, A., Jia, Y., and He, K. Accurate, large minibatch SGD: training imagenet in 1 hour, 2017.

Gu, A. and Dao, T. Mamba: Linear-time sequence modeling with selective state spaces. *CoRR*, abs/2312.00752, 2023.

Gu, A., Goel, K., and Ré, C. Efficiently modeling long sequences with structured state spaces. *ArXiv*, 2111.00396, 2021.

Gupta, A., Gu, A., and Berant, J. Diagonal state spaces are as effective as structured state spaces. *ArXiv*, 2203.14343, 2022.

Hatamizadeh, A., Tang, Y., Nath, V., Yang, D., Myronenko, A., Landman, B. A., Roth, H. R., and Xu, D. UNETR: transformers for 3d medical image segmentation. In *IEEE/CVF Winter Conference on Applications of Computer Vision, WACV 2022, Waikoloa, HI, USA, January 3-8, 2022*, pp. 1748–1758. IEEE, 2022.

He, K., Chen, X., Xie, S., Li, Y., Dollár, P., and Girshick, R. B. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022*, pp. 15979–15988, 2022.

Helber, P., Bischke, B., Dengel, A., and Borth, D. Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE J. Sel. Top. Appl. Earth Obs. Remote. Sens.*, 12(7):2217–2226, 2019.

Hendrycks, D. and Dietterich, T. G. Benchmarking neural network robustness to common corruptions and perturbations. In *ICLR (Poster)*. OpenReview.net, 2019.

Hendrycks, D., Basart, S., Mu, N., Kadavath, S., Wang, F., Dorundo, E., Desai, R., Zhu, T., Parajuli, S., Guo, M., Song, D., Steinhardt, J., and Gilmer, J. The many faces of robustness: A critical analysis of out-of-distribution generalization. *ICCV*, 2021a.

Hendrycks, D., Zhao, K., Basart, S., Steinhardt, J., and Song, D. Natural adversarial examples. *CVPR*, 2021b.

Huang, G., Sun, Y., Liu, Z., Sedra, D., and Weinberger, K. Q. Deep networks with stochastic depth. In Leibe, B., Matas, J., Sebe, N., and Welling, M. (eds.), *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV*, volume 9908 of *Lecture Notes in Computer Science*, pp. 646–661. Springer, 2016.

Johnson, J., Hariharan, B., van der Maaten, L., Fei-Fei, L., Zitnick, C. L., and Girshick, R. B. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pp. 1988–1997. IEEE Computer Society, 2017.

Kaggle and EyePacs. Kaggle diabetic retinopathy detection, July 2015.

Kalamkar, D. D., Mudigere, D., Mellempudi, N., Das, D., Banerjee, K., Avancha, S., Vooturi, D. T., Jammalamadaka, N., Huang, J., Yuen, H., Yang, J., Park, J., Heinecke, A., Georganas, E., Srinivasan, S., Kundu, A., Smelyanskiy, M., Kaul, B., and Dubey, P. A study of BFLOAT16 for deep learning training. *CoRR*, abs/1905.12322, 2019.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In Bengio, Y. and LeCun, Y. (eds.), *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A. C., Lo, W., Dollár, P., and Girshick, R. B. Segment anything. In *ICCV*, pp. 3992–4003. IEEE, 2023.

Krizhevsky, A. Learning multiple layers of features from tiny images. 2009.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In Bartlett, P. L., Pereira, F. C. N., Burges, C. J. C., Bottou, L., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, pp. 1106–1114, 2012.

Liu, Y., Tian, Y., Zhao, Y., Yu, H., Xie, L., Wang, Y., Ye, Q., and Liu, Y. Vmamba: Visual state space model. *CoRR*, abs/2401.10166, 2024.

Liu, Z., Mao, H., Wu, C., Feichtenhofer, C., Darrell, T., and Xie, S. A convnet for the 2020s. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition,*

CVPR 2022, New Orleans, LA, USA, June 18-24, 2022, pp. 11966–11976. IEEE, 2022.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net, 2019.

Nguyen, T., Brandstetter, J., Kapoor, A., Gupta, J. K., and Grover, A. Climax: A foundation model for weather and climate. *arXiv preprint arXiv:2301.10343*, 2023.

Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., Assran, M., Ballas, N., Galuba, W., Howes, R., Huang, P., Li, S., Misra, I., Rabbat, M. G., Sharma, V., Synnaeve, G., Xu, H., Jégou, H., Mairal, J., Labatut, P., Joulin, A., and Bojanowski, P. Dinov2: Learning robust visual features without supervision. *CoRR*, abs/2304.07193, 2023.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E. Z., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pp. 8024–8035, 2019.

Peebles, W. and Xie, S. Scalable diffusion models with transformers. In *IEEE/CVF International Conference on Computer Vision, ICCV 2023, Paris, France, October 1-6, 2023*, pp. 4172–4182. IEEE, 2023.

Singh, M., Duval, Q., Alwala, K. V., Fan, H., Aggarwal, V., Adcock, A., Joulin, A., Dollár, P., Feichtenhofer, C., Girshick, R. B., Girdhar, R., and Misra, I. The effectiveness of MAE pre-pretraining for billion-scale pretraining. *CoRR*, abs/2303.13496, 2023.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., and Wojna, Z. Rethinking the inception architecture for computer vision. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 2818–2826. IEEE Computer Society, 2016.

Team, G., Anil, R., Borgeaud, S., Wu, Y., Alayrac, J.-B., Yu, J., Soricut, R., Schalkwyk, J., Dai, A. M., Hauth, A., et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Touvron, H., Cord, M., Douze, M., Massa, F., Sablayrolles, A., and Jégou, H. Training data-efficient image transformers & distillation through attention. In *ICML*, volume 139 of *Proceedings of Machine Learning Research*, pp. 10347–10357. PMLR, 2021a.

Touvron, H., Cord, M., Sablayrolles, A., Synnaeve, G., and Jégou, H. Going deeper with image transformers. In *ICCV*, pp. 32–42. IEEE, 2021b.

Touvron, H., Cord, M., El-Nouby, A., Verbeek, J., and Jégou, H. Three things everyone should know about vision transformers. In *ECCV (24)*, volume 13684 of *Lecture Notes in Computer Science*, pp. 497–515. Springer, 2022a.

Touvron, H., Cord, M., and Jégou, H. Deit III: revenge of the vit. In *ECCV (24)*, volume 13684 of *Lecture Notes in Computer Science*, pp. 516–533. Springer, 2022b.

Valanarasu, J. M. J., Oza, P., Hacihaliloglu, I., and Patel, V. M. Medical transformer: Gated axial-attention for medical image segmentation. In de Bruijne, M., Cattin, P. C., Cotin, S., Padoy, N., Speidel, S., Zheng, Y., and Essert, C. (eds.), *Medical Image Computing and Computer Assisted Intervention - MICCAI 2021 - 24th International Conference, Strasbourg, France, September 27 - October 1, 2021, Proceedings, Part I*, volume 12901 of *Lecture Notes in Computer Science*, pp. 36–46. Springer, 2021.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Veeling, B. S., Linmans, J., Winkens, J., Cohen, T., and Welling, M. Rotation equivariant cnns for digital pathology. In Frangi, A. F., Schnabel, J. A., Davatzikos, C., Alberola-López, C., and Fichtinger, G. (eds.), *Medical Image Computing and Computer Assisted Intervention - MICCAI 2018 - 21st International Conference, Granada, Spain, September 16-20, 2018, Proceedings, Part II*, volume 11071 of *Lecture Notes in Computer Science*, pp. 210–218. Springer, 2018.

Wang, F., Wang, J., Ren, S., Wei, G., Mei, J., Shao, W., Zhou, Y., Yuille, A., and Xie, C. Mamba-r: Vision mamba also needs registers. *arXiv preprint arXiv:2405.14858*, 2024.

Wang, H., Ge, S., Lipton, Z., and Xing, E. P. Learning robust global representations by penalizing local predictive power. In *Advances in Neural Information Processing Systems*, pp. 10506–10518, 2019.

Xiao, J., Hays, J., Ehinger, K. A., Oliva, A., and Torralba, A. SUN database: Large-scale scene recognition from abbey to zoo. In *The Twenty-Third IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2010, San Francisco, CA, USA, 13-18 June 2010*, pp. 3485–3492. IEEE Computer Society, 2010.

Xiao, T., Liu, Y., Zhou, B., Jiang, Y., and Sun, J. Unified perceptual parsing for scene understanding. In Ferrari, V., Hebert, M., Sminchisescu, C., and Weiss, Y. (eds.), *Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part V*, volume 11209 of *Lecture Notes in Computer Science*, pp. 432–448. Springer, 2018.

Xu, H., Usuyama, N., Bagga, J., Zhang, S., Rao, R., Naumann, T., Wong, C., Gero, Z., González, J., Gu, Y., et al. A whole-slide foundation model for digital pathology from real-world data. *Nature*, pp. 1–8, 2024.

Yun, S., Han, D., Chun, S., Oh, S. J., Yoo, Y., and Choe, J. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pp. 6022–6031. IEEE, 2019.

Zhai, X., Puigcerver, J., Kolesnikov, A., Ruyssen, P., Riquelme, C., Lucic, M., Djolonga, J., Pinto, A. S., Neumann, M., Dosovitskiy, A., et al. A large-scale study of representation learning with the visual task adaptation benchmark. *arXiv preprint arXiv:1910.04867*, 2019.

Zhang, H., Cissé, M., Dauphin, Y. N., and Lopez-Paz, D. mixup: Beyond empirical risk minimization. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net, 2018.

Zhou, B., Zhao, H., Puig, X., Xiao, T., Fidler, S., Barriuso, A., and Torralba, A. Semantic understanding of scenes through the ADE20K dataset. *Int. J. Comput. Vis.*, 127 (3):302–321, 2019.

Zhu, L., Liao, B., Zhang, Q., Wang, X., Liu, W., and Wang, X. Vision mamba: Efficient visual representation learning with bidirectional state space model. *CoRR*, abs/2401.09417, 2024.

A. Extended Results

A.1. Vision Adjustments of xLSTM

The xLSTM (Beck et al., 2024) architecture was designed for language modeling and therefore includes some design choices that are specific to language modeling. One of these design choices is a causal 1D convolution (with kernel size 4) to encode local context into the query and key input of the mLSTM layer. As images are neither causal nor 1D structures, we replace the causal 1D convolution with a 2D convolution (with kernel size 3). When training a ViL-T for 400 epochs, this improves ImageNet-1K accuracy from 76.1% to 76.7%.

Another language specific design choice is the absence of bias parameters in projection and layernorm (Ba et al., 2016) layers which improves training stability for larger models (Chowdhery et al., 2023; Dehghani et al., 2023). As vision models are typically much smaller than language models, ViTs commonly use all bias parameters. When adding bias parameters to the ViL architecture, a ViL-T trained for 400 epochs improves from 76.7% to 77.1% in ImageNet-1K classification.

A.2. ADE20K Semantic Segmentation

Table 4 shows results for transferring ImageNet-1K pre-trained models to ADE20K (Zhou et al., 2019) semantic segmentation using UperNet (Xiao et al., 2018). Also here, ViL shows strong performances across the board, even outperforming DeiT-III-B despite the lower ImageNet-1K accuracy of ViL-B.

For this benchmark, we train on 512x512 images and inference could be conducted on the original images (without resizing), which results in even larger images. Consequently, the number of patch tokens can become quite large. As sequence length increases, ViL would ideally use a recurrent or chunkwise formulation of the mLSTM. Shortly summarized, the mLSTM can operate in parallel, recurrent or chunked modes, each with distinct FLOPS and runtime characteristics. Given a sequence length L and hidden dimension D, the trade-off between modes can be formulated as $\mathcal{O}(LCD + LD^2)$ where C denotes the chunksize of the chunked form. $C = 1$ (recurrent mode) yields complexity $\mathcal{O}(LD^2)$ and $C = L$ (parallel mode) gives $\mathcal{O}(L^2D)$ or $\mathcal{O}(L\frac{L}{2}D)$ when omitting the causally masked operations. As there currently does not exist an optimized hardware implementation for the mLSTM, it is difficult to estimate runtime or FLOPS that are of practical relevance in this setting and therefore leave it to future work.

		Single-scale		**Multi-scale**	
Model	#Params	mIoU	ACC	mIoU	ACC
DeiT-T (Touvron et al., 2021a)	10M	38.1	78.2	40.3	79.9
DeiT-III-T (Touvron et al., 2022b)	10M	39.8	79.2	42.2	80.7
Vim-T (Zhu et al., 2024)	13M	41.0	-	-	-
ViL-T	11M	**41.2**	**80.2**	**43.1**	**81.3**
DeiT-S (Touvron et al., 2021a)	41M	43.1	80.7	45.2	81.8
DeiT-III-S (Touvron et al., 2022b)	41M	45.2	81.5	46.3	82.3
Vim-S (Zhu et al., 2024)	46M	44.9	-	-	-
Mamba®-S (Wang et al., 2024)	56M	45.3	-	-	-
ViL-S	42M	**46.3**	**82.0**	**47.9**	**82.9**
DeiT-B (Touvron et al., 2021a)	113M	45.8	82.1	47.0	82.9
DeiT-III-B (Touvron et al., 2022b)	113M	47.5	82.6	49.0	83.3
Mamba®-B (Wang et al., 2024)	132M	47.7	-	-	-
ViL-B	115M	**48.6**	**82.8**	**49.6**	**83.3**

Table 4. Semantic segmentation results on ADE20K (Zhou et al., 2019) using UperNet (Xiao et al., 2018). We report mean intersection over union (mIoU) and pixelwise accuracy (ACC) for single- and multi-scale evaluation. Models are trained for 160K updates with a batchsize of 16 on 512x512 resolution. Detailed hyperparameters are listed in Appendix Table 12.

A.3. VTAB-1K Transfer Classification

Table 5 shows transfer classification results for ImageNet-1K pre-trained models on the VTAB-1K (Zhai et al., 2019) benchmark. VTAB-1K consists of 19 datasets split into 7 natural datasets (such as CIFAR100 (Krizhevsky, 2009), Caltech101 (Fei-Fei et al., 2006) or SUN397 (Xiao et al., 2010)), 4 specialized datasets (medical imaging (Veeling et al.,

2018; Kaggle & EyePacs, 2015) and remote sensing (Helber et al., 2019; Cheng et al., 2017)) and 8 structured datasets (with tasks such as object counting (Johnson et al., 2017) or binned depth estimation (Geiger et al., 2013)). We follow common practices and tune the learning rate per model and dataset on the validation set followed by training with the best learning rate on the union of train and validation set. The performance metric is the average testset accuracy over 5 seeds. ViL shows strong transfer classification performance outperforming all other models on the average over all 19 datasets. ViL performs particularly well on the structured datasets where ViL-B outperforms DeiT-III-B despite ViL-B having lower ImageNet-1K accuracy.

Model	#Params	FLOPS	Natural	Specialized	Structured	Average
DeiT-T (Touvron et al., 2021a)	6M	1.3G	69.2	82.0	53.3	65.2
DeiT-III-T (reimpl.)	6M	1.3G	71.9	82.6	55.2	67.1
Vim-T (Zhu et al., 2024)	7M	1.5G	68.0	80.7	47.1	61.9
ViL-T	6M	1.5G	**73.6**	**83.4**	**56.1**	**68.3**
DeiT-S (Touvron et al., 2021a)	22M	4.6G	73.3	83.8	53.2	67.1
DeiT-III-S (Touvron et al., 2022b)	22M	4.6G	75.0	83.2	52.3	67.2
Vim-S (Zhu et al., 2024)	26M	5.3G	69.6	81.7	49.4	63.6
ViL-S	23M	5.1G	**75.3**	**84.3**	**58.3**	**70.0**
DeiT-B (Touvron et al., 2021a)	86M	17.6G	76.5	**85.2**	55.7	69.6
DeiT-III-B (Touvron et al., 2022b)	86M	17.6G	**77.6**	84.8	56.6	70.3
ViL-B	89M	18.6G	76.6	84.7	**59.1**	**70.9**

Table 5. Transfer classification accuracies on the VTAB-1K (Zhai et al., 2019) benchmark using ImageNet-1K pre-trained models. VTAB-1K consists of 19 datasets split into 7 natural, 4 specialized and 8 structured datasets. We show averages per category and the average accuracy over all 19 datasets (Appendix Table 9 lists all individual accuracies). ViL shows strong transfer classification performance, outperforming heavily optimized ViT protocols and Vim on the full VTAB-1K benchmark. Notably, ViL performs exceptionally well on the structured category. We tune the learning rate for each model and dataset on the validation set and report the average performance on the test set over 5 seeds. Appendix Table 13 lists further hyperparameters.

A.4. Runtime Comparison of ViL vs Vim

We compare the time it takes to train ViL and Vim (Zhu et al., 2024) for a single ImageNet-1K epoch in Table 6. We follow the scaling procedure of ViTs, using 192 (T), 384 (S), 768 (B), 1024 (L) as latent dimension where the large scale additionally doubles the number of blocks.

Model	Optimization	(T)iny	(S)mall	(B)ase	(L)arge
Vim (Zhu et al., 2024)	custom CUDA kernel	7.3h	14.0h	28.2h	76.4h
ViL	`torch.compile`	5.0h	8.7h	16.6h	45.1h
Speedup of ViL compared to Vim		45%	61%	69%	69%

Table 6. Runtime comparisons between Vim (Zhu et al., 2024) and ViL. ViL is up to 69% faster despite the current lack of a custom CUDA kernel. As mLSTM (and ViL) can be parallelized analogous to FlashAttention (Dao et al., 2022; Dao, 2023) via custom hardware optimizations, ViL will become even faster in the future. Runtimes denote the training time for 10 ImageNet-1K epochs and are extrapolated from short benchmark runs on a single A100-80GB-PCIe using float16 precision and 224x224 images.

A.5. Impact of Longer Training

We investigate the impact of training for a longer duration in Table 7. We also attempted to train larger models for longer but did not observe similar performance gains. One could probably change that by tuning hyperparameters for an 800 epoch training schedule, which we omit due to the large computational costs associated with training larger models.

A.6. Long-sequence fine-tuning

Table 8 shows results for the long-sequence fine-tuning setting of Vim (Zhu et al., 2024).

Model	Epochs	IN-1K ACC	VTAB-1K	ADE20K mIoU
DeiT-III-T	400	75.6	67.0	39.1
DeiT-III-T	800	76.2	67.1	39.8
ViL-T	400	77.2	67.8	40.9
ViL-T	800	78.3	68.3	41.2

Table 7. Performance comparison of tiny models when training for 400 and 800 epochs. ADE20K mIoU uses single-scale evaluation. All settings follow the ones used in the main paper.

Method	#Params	ACC
DeiT-III-T (reimpl.)	6M	78.3
Vim-T (Zhu et al., 2024)	7M	78.3
ViL-T	6M	**80.1**
DeiT-S (Touvron et al., 2021a)	22M	81.4
DeiT-III-S (Touvron et al., 2022b)	22M	**83.2**
Vim-S (Zhu et al., 2024)	26M	81.6
ViL-S	23M	82.7
DeiT-B (Touvron et al., 2021a)	86M	83.2
DeiT-III-B (Touvron et al., 2022b)	86M	**84.5**
ViL-B	89M	83.3

Table 8. Fine-tuning ImageNet-1K pre-trained models by using a 50% overlap when creating patches. This setting increases the number of patch tokens from 196 to 729. ViL easily outperforms Vim but falls short of ViTs on larger models. Note that ViL FLOPS are calculated using the parallel form of the mLSTM, which scales quadratically with the sequence length. Using the chunkwise form instead of the parallel form can reduce FLOPS for longer sequences (see Section B.2).

A.7. VTAB-1K Individual Dataset Results

Table 9 presents accuracies for each individual dataset of the VTAB-1K (Zhai et al., 2019) benchmark.

	Natural							**Specialized**				**Structured**							
	Cifar100	Caltech101	DTD	Flower102	Pets	SVHN	Sun397	Camelyon	EuroSAT	Resisc45	Retinopathy	Clevr-Count	Clevr-Dist	DMLab	KITTI-Dist	dSpr-Loc	dSpr-Ori	sNORB-Azim	sNORB-Ele
DeiT-T (Touvron et al., 2021a)	47.7	86.4	63.7	85.6	87.0	78.4	35.3	83.0	93.4	80.9	70.7	71.7	60.3	43.1	78.5	**67.9**	41.6	30.6	32.7
DeiT-III-T (reimpl.)	52.3	90.1	62.7	88.8	87.5	**83.7**	37.9	83.2	93.1	81.1	72.9	76.6	60.8	44.9	79.1	67.5	48.1	31.0	33.3
Vim-T (Zhu et al., 2024)	46.7	86.3	60.7	84.0	88.8	76.1	33.7	82.2	92.9	75.2	72.6	59.8	49.9	39.3	78.2	51.2	43.9	26.9	27.2
ViL-T	**54.2**	**90.2**	**67.4**	**90.7**	**89.9**	81.6	**41.1**	**83.4**	**94.2**	**82.7**	**73.1**	**80.7**	**61.8**	**49.4**	**81.3**	57.8	**51.8**	**31.4**	**34.8**
DeiT-S (Touvron et al., 2021a)	57.0	88.9	68.2	90.9	90.8	75.4	42.1	83.3	**94.0**	83.8	**74.0**	74.6	58.3	45.6	78.2	**61.9**	47.9	27.1	31.9
DeiT-III-S (Touvron et al., 2022b)	58.8	88.6	67.5	90.9	**91.7**	**84.4**	43.3	84.4	92.6	82.5	73.5	76.5	57.9	46.2	78.9	58.3	49.7	23.7	27.5
Vim-S (Zhu et al., 2024)	53.0	87.2	64.1	86.8	90.3	65.8	39.7	82.4	93.4	78.0	73.1	63.1	53.2	42.3	78.2	54.1	47.6	27.1	29.3
ViL-S	**61.4**	**89.6**	**69.2**	**92.8**	**91.7**	78.7	**43.8**	**85.5**	93.9	**84.4**	73.5	**84.0**	**63.4**	**51.3**	**83.3**	61.0	**55.4**	**32.4**	**35.5**
DeiT-B (Touvron et al., 2021a)	61.8	89.8	67.5	**93.7**	92.6	84.4	45.6	85.3	**95.1**	**86.3**	74.2	77.7	59.9	47.2	**81.7**	61.7	51.4	30.0	36.2
DeiT-III-B (Touvron et al., 2022b)	62.9	89.6	69.6	**93.7**	**93.2**	**87.0**	**47.1**	**85.8**	94.1	85.6	73.7	80.5	61.4	48.4	80.9	**64.4**	55.1	30.2	31.8
ViL-B	**64.3**	**90.0**	**71.1**	93.4	91.4	79.6	46.6	84.7	94.3	85.4	**74.4**	**83.7**	**62.1**	**52.7**	81.0	63.1	**57.6**	**32.6**	**39.9**

Table 9. Results on all datasets of the VTAB-1K (Zhai et al., 2019) benchmark.

A.8. Robustness

Table 10 presents robustness and OOD evaluations of ImageNet-1K pre-trained classifiers.

B. Implementation Details

B.1. Hardware

We train models on a mix of custom hardware servers (mainly A100 and A40 GPUs) and public research clusters equiped with either 8xA100 or 4xA100 nodes.

Model	IN-C (↓)	IN-A (↑)	IN-R (↑)	Sketch (↑)	Validation (↑)
DeiT-T (Touvron et al., 2021a)	69.7	7.6	32.7	19.9	72.2
DeiT-III-T	65.0	11.7	39.4	27.4	76.2
Vim-T (Zhu et al., 2024)	61.8	9.6	38.8	26.9	76.1
ViL-T	**59.6**	**15.2**	**42.2**	**30.0**	**78.3**
DeiT-S (Touvron et al., 2021a)	54.4	19.6	41.9	29.1	79.8
DeiT-III-S (Touvron et al., 2022b)	**50.1**	23.2	46.6	**35.4**	81.4
Vim-S (Zhu et al., 2024)	51.5	19.7	44.8	32.5	80.5
ViL-S	50.6	**23.8**	**47.9**	35.2	**81.5**
DeiT-B (Touvron et al., 2021a)	48.6	27.9	44.6	32.0	81.8
DeiT-III-B (Touvron et al., 2022b)	**42.7**	**36.5**	**54.1**	**41.1**	**83.8**
ViL-B	45.3	30.9	51.9	39.0	82.4

Table 10. Robustness and OOD evaluations on ImageNet-C(orruption) (Hendrycks & Dietterich, 2019), ImageNet-A(dversarial) (Hendrycks et al., 2021b), ImageNet-R(endition) (Hendrycks et al., 2021a) and ImageNet-Sketch (Wang et al., 2019).. For ImageNet-C, we report the mean corruption error (Hendrycks & Dietterich, 2019) with AlexNet (Krizhevsky et al., 2012) as baseline.

We estimate the total number of A100 GPU-hours used for this project to be 38K hours. This estimate includes initial exploration, method development, analysis and evaluations.

B.2. FLOPS Calculation

We calculate the FLOPS with the `fvcore` library[1]. For ViL, we report the FLOPS of the mLSTM parallel form. Given sequence length L and hidden dimension D, it has complexity $\mathcal{O}(L\frac{L}{2}D)$ where the $\frac{L}{2}$ stems from the causal masking.

We report FLOPS for a complexity of $\mathcal{O}(L\frac{L}{2}D)$ even though our current implementation of the mLSTM has complexity $\mathcal{O}(L^2D)$ (i.e., entries of the QK matrix that are later masked are still calculated) which we justify by the fact that FlashAttention-2 (Dao, 2023) is approximately 1.7x faster with a causal mask than without. Therefore, an optimized hardware implementation of the mLSTM could also omit the calculation of values that are not needed.

As Vim (Zhu et al., 2024) does not report FLOPS and their model makes use of CUDA kernels (which are not counted as FLOPS by `fvcore`), we replace all calls to CUDA kernels with their reference PyTorch implementation and count the FLOPS with `fvcore`.

For the total pre-training compute in Figure 2, we consider an efficient implementation of stochastic depth (Huang et al., 2016) which omits the calculation of a dropped block instead of masking it. Therefore, we change the implementation of ViT (Dosovitskiy et al., 2021) and ConvNeXt (Liu et al., 2022) to use our efficient stochastic depth implementation. Vim does not use stochastic depth for training as their models are relatively small.

[1] https://github.com/facebookresearch/fvcore

B.3. ViL Hyperparameters

Table 11 shows detailed hyperparameters used to train ViL models.

Parameter	Value
Epochs	800 (T), 400 (S/B) → 20 (T, S), 5 (B)
Batch size	2048 → 1024
Model	
Patch size	16x16
Latent dimension	192 (T), 384 (S), 768 (B)
Depth	24
Pooling	Bilateral Concat
Stochastic depth (Huang et al., 2016)	
Peak rate	0 (T), 0.05 (S), 0.2 (B)
Layer-wise Decay	✗
Optimizer	AdamW (Loshchilov & Hutter, 2019; Kingma & Ba, 2015)
Base Learning rate	1e-3 → 1e-5
Linear LR Scaling Divisor (Goyal et al., 2017)	1024
Weight decay	0.05
Momentum	$\beta_1 = 0.9, \beta_2 = 0.999$
Gradient Norm Clip	1.0
Precision	mixed `bfloat16` (Kalamkar et al., 2019)
Backend	`torch.autocast`
Learning rate schedule	cosine decay
Warmup schedule	linear
Warmup epochs	5 → 5 (T, S), 1 (B)
End LR	1e-6
Label smoothing (Szegedy et al., 2016)	✗
Train Data Augmentation	
RandomResizedCrop	192 → 224
Scale	[0.08, 1.0]
Interpolation	bicubic
RandomHorizontalFlip	$p = 0.5$
3-Augment (Touvron et al., 2022b)	
Gaussian Blur σ	[0.1, 2.0]
ColorJitter	[0.3, 0.3, 0.3, 0.0]
Normalize	ImageNet-1K statistics
Mixup (Zhang et al., 2018) α	0.8
Cutmix (Yun et al., 2019) α	1.0
Test Data Augmentation	
Resize	192 → 224
Interpolation	bicubic
CenterCrop	192 → 224
Normalize	ImageNet-1K statistics

Table 11. Hyperparameters for training ViL on ImageNet-1K, inspired by DeiT-III (Touvron et al., 2022b). We follow the best setting from DeiT-III (Touvron et al., 2022b) and pre-train on 192 resolution followed by a short fine-tuning on 224 resolution (indicated by →).

B.4. ADE20K Semantic Segmentation Fine-tuning

We fine-tune models on ADE20K (Zhou et al., 2019) using an UperNet (Xiao et al., 2018) head. We follow common practices and fine-tune on 512x512 resolution, where we interpolate the absolute positional embedding from 224x224 to 512x512. For ViTs, we add relative position biases to the attention layers (initialized to 0) (He et al., 2022). Table 12 lists detailed hyperparameters.

Parameter	Value
Updates	160K
Batch size	16
UperNet	
Auxiliary	
Weight	0.4
Input Block	8*
Dimension	192 (T), 384 (S, B)
Decoder	
Weight	1.0
Input Blocks	[4, 6, 8, 12]*
Dimension	192 (T), 384 (S, B)
Stochastic depth (Huang et al., 2016)	
Peak rate	0 (T), 0.05 (S), 0.1 (B)
Layer-wise Decay	✓
Optimizer	AdamW (Loshchilov & Hutter, 2019; Kingma & Ba, 2015)
Learning rate	5e-4
Linear LR Scaling Divisor (Goyal et al., 2017)	16
Layer-wise lr decay (Clark et al., 2020)	0.65*
Weight decay	0.05
Momentum	$\beta_1 = 0.9, \beta_2 = 0.999$
Learning rate schedule	linear warmup $\rightarrow$ cosine decay
Warmup updates	1500
Precision	mixed `float16`
Backend	`torch.autocast`
Train Data Augmentation	
`RandomResize`	
`interpolation`	bicubic
`RandomCrop`	
`size`	512x512
`RandomHorizontalFlip`	
`ColorJitter`	0.5
`brightness`	0.5
`contrast`	0.5
`saturation`	0.5
`hue`	0.25
`Normalize`	ImageNet-1K statistics
Evaluation	
`Stride`	341
`Multi-scale`	
`scale factors`	[0.75, 1.0, 1.25, 1.5, 1.75]
`flip`	[True, False]

Table 12. Hyperparameters for fine-tuning on VTAB-1K. *For ViL we group two consecutive blocks into one similar to how a ViT block consists of a pair of attention and MLP block.

B.5. Fine-tuning on VTAB-1K

For fine-tuning models on VTAB-1K we provide the hyperparameters in Table 13. We search for the best learning rate for each dataset by fine-tuning the model 25 times (5 learning rates with 5 seeds each) on the 800 training samples and evaluating them on the 200 validation samples. With the best learning rate, we then train each model 5 times on concatenation of training and validation split, evaluate on the test split and report the average accuracy.

Parameter	Value
Epochs	50
Batch size	64
Seeds	5
Optimizer	AdamW (Loshchilov & Hutter, 2019; Kingma & Ba, 2015)
Learning rate	[1e-3, 7.5e-4, 5.0e-4, 2.5e-4, 1.0e-4]
Layer-wise lr decay (Clark et al., 2020)	0.65*
Weight decay	0.05
Momentum	$\beta_1 = 0.9, \beta_2 = 0.999$
Learning rate schedule	linear warmup $\rightarrow$ cosine decay
Warmup epochs	5
Precision	mixed `bfloat16` (Kalamkar et al., 2019)
Backend	`torch.autocast`
Data Augmentation	
`Resize`	
`interpolation`	bicubic
`size`	224x224
`Normalize`	ImageNet-1K statistics

Table 13. Hyperparameters for fine-tuning on VTAB-1K. *For Vim and ViL we group two consecutive blocks for the layer-wise lr decay similar to how ViT considers a pair of attention and MLP block as a single "layer" for the decay.

B.6. DeiT-III Reimplementation Hyperparameters

Table 11 shows detailed hyperparameters used to train DeiT-III-T (reimpl.) from Table 1.

Parameter	Value
Epochs	800 → 20
Batch size	2048 → 1024
Model	
Patch size	16x16
Latent dimension	192
Depth	12
Pooling	[CLS]
Stochastic depth (Huang et al., 2016)	✗
Layerscale (Touvron et al., 2021b)	1e-4
Optimizer	AdamW (Loshchilov & Hutter, 2019; Kingma & Ba, 2015)
Base Learning rate	1e-3 → 1e-5
Linear LR Scaling Divisor (Goyal et al., 2017)	1024
Weight decay	0.05
Momentum	$\beta_1 = 0.9, \beta_2 = 0.999$
Gradient Norm Clip	✗
Precision	mixed `bfloat16` (Kalamkar et al., 2019)
Backend	`torch.autocast`
Learning rate schedule	cosine decay
Warmup schedule	linear
Warmup epochs	5
End LR	1e-6
Label smoothing (Szegedy et al., 2016)	✗
Train Data Augmentation	
RandomResizedCrop	192 → 224
Scale	[0.08, 1.0]
Interpolation	bicubic
RandomHorizontalFlip	$p = 0.5$
3-Augment (Touvron et al., 2022b)	
Gaussian Blur σ	[0.1, 2.0]
ColorJitter	[0.3, 0.3, 0.3, 0.0]
Normalize	ImageNet-1K statistics
Mixup (Zhang et al., 2018) α	0.8
Cutmix (Yun et al., 2019) α	1.0
Test Data Augmentation	
Resize	192 → 224
Interpolation	bicubic
CenterCrop	192 → 224
Normalize	ImageNet-1K statistics

Table 14. Hyperparameters for training our reimplementation of DeiT-III-T (Touvron et al., 2022b) on ImageNet-1K. The most significant change is that we reduce the learning rate from 3e-3 to 1e-3 as we found this to greatly improve performance. We make minor changes to the protocol such as using AdamW or no gradient clipping as models were stable without it. We follow the best setting from DeiT-III (Touvron et al., 2022b) and pre-train on 192 resolution followed by a short fine-tuning on 224 resolution (indicated by →).