Contextual Vision Transformers for Robust Representation Learning

Yujia Bao [1] **Theofanis Karaletsos** [1]

Abstract

We present Contextual Vision Transformers (ContextViT), a method for producing robust feature representations for images exhibiting grouped structure such as covariates. ContextViT introduces an extra context token to encode group-specific information, allowing the model to explain away group-specific covariate structures while keeping core visual features shared across groups. Specifically, given an input image, Context-ViT maps images that share the same covariate into this context token appended to the input image tokens to capture the effects of conditioning the model on group membership. We furthermore introduce a context inference network to predict such tokens on the fly given a few samples from a group distribution, enabling ContextViT to generalize to new testing distributions at inference time. We demonstrate the performance of ContextViT through a diverse range of applications.

1. Introduction

In recent years, Vision Transformers (ViTs) have emerged as a powerful approach for image representation learning (Dosovitskiy et al.; Khan et al., 2022). However, many real-world datasets exhibit structured variations or covariate shifts across groups, which pose challenges for standard ViT models. To address this limitation, we propose Contextual Vision Transformers (ContextViT), a novel method that leverages the notion of in-context learning (Brown et al., 2020) to produce robust feature representations for images with grouped structures, such as covariates.

ContextViT builds upon the concept of in-context learning, which was originally proposed in the domain of language processing. In-context learning involves conditioning Transformer-based models on specific contexts to enable test-time adaptation to new tasks. In our work, we extend this idea to address the challenge of *covariate shift* and *structured variations* in real-world datasets. Unlike traditional in-context learning approaches that condition on the whole context-dataset explicitly for each query, ContextViT infers context tokens per each group that can be shared across members of this group, such as all images collected under a shared context. A practical advantage of this modeling style is not requiring extensive memory and computing resources to serve excessive context windows. Underlying this model is the idea of *mixed models*, a class of hierarchical statistical models commonly used in scientific applications, which capture fixed and random effects to represent individual- and group-level variation for observed data, but in our case those effects are latent and modeled as inferred context tokens. In-context learning is also interpreted as an implicit generative model (Xie et al., 2022), but in our case the relatedness of the queries is explicitly given by the group structure, assuming a more structured style of conditioning across contexts common in real world scientific datasets collected under shared experimental conditions.

We present two variations of ContextViT: In Section 2.3, we assume contexts have been observed and infer them as parameters during training. Test-time adaptation is impractical for this model as it requires inferring a new token for a test group. In Section 2.4, we propose overcoming this limitation by utilizing a *context inference network* to estimate these tokens on the fly given a set of examples from a group, making the model more amenable to test-time adaptation tasks. We examine different choices for the inference network and *layerwise* versus input-layer-only inference of context tokens.

We demonstrate the versatility and effectiveness of ContextViT and its variations on the challenging Camelyon17 dataset, which serves as a benchmark for distribution shift in pathology imaging. Compared to conventional ViT models, ContextViT showcases remarkable improvements in generalization, achieving significantly higher accuracy on previously unseen hospitals (97.5 compared to 93.8).

2. Contextual Vision Transformer

Assume $\mathcal{D}_c$ denotes a collection of samples from a distribution $P_c(\cdot)$ under potentially unknown context c. Our goal is as follows: given a collection of such related datasets from

[1]Insitro. Correspondence to: Yujia Bao <yujia@insitro.com>, Theofanis Karaletsos <theofanis@insitro.com>.

Published at the ICML 2023 Workshop on Spurious Correlations, Invariance, and Stability. Baltimore, Maryland, USA. Copyright 2023 by the author(s).

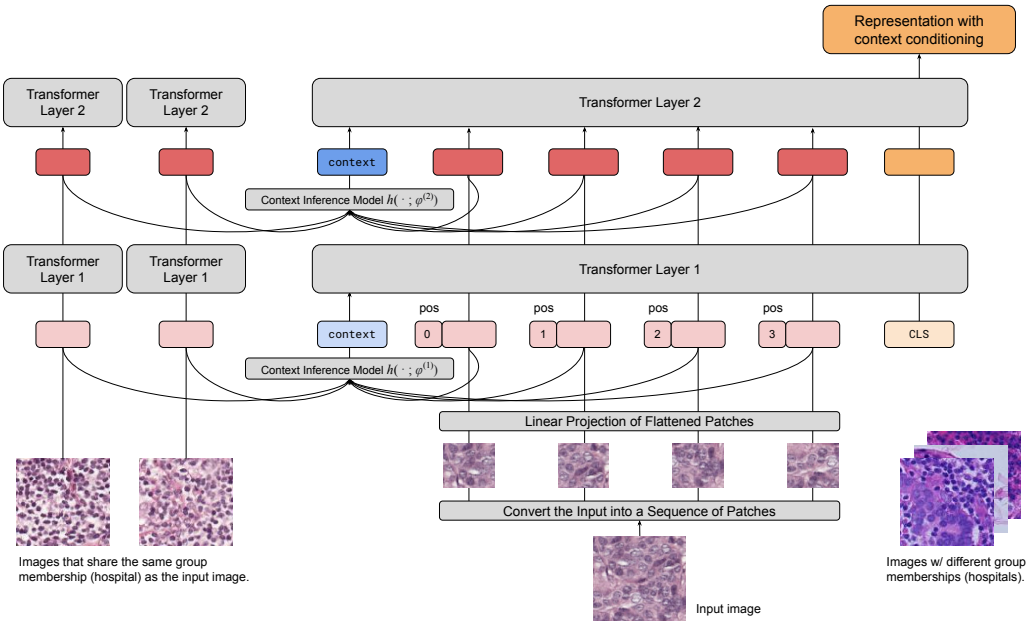

Figure 1. Contextual Vision Transformer (ContextViT). Before each Transformer layer, ContextViT applies a context inference model over all patches (from images within the current batch that share the same group membership) to produce a context token. To simplify the illustration, we present only two Transformer layers and depict a single patch embedding for each image on the left side.

distributions with varying contexts c, we would like to learn a model that can generalize gracefully across these different domains and ideally adapt to new contexts c^* at test time.

We assume that we observe data from multiple distributions jointly, each varying by a context c, so that our dataset is comprised by their collection $\mathcal{D} = \{\mathcal{D}_1, ..., \mathcal{D}_c\}$. We aim to learn a single shared ViT model, parameterized by θ, across all sub-datasets with varying contexts. We denote a tuple $\{x, y, c\}$ as the data describing the sample, where x denotes an input, the image, and y denotes an output variable such as a label or an unknown embedding, and c denoting group membership for each datum to each sub-distribution $P_c(\cdot)$. Over the next sections we will explain our assumptions for ContextViT and will present practical implementations.

2.1. In-Context Model

A popular paradigm for incorporating test-time conditioning in Transformers is given by in-context learning (Brown et al., 2020) and prompt-conditioning (Radford et al., 2021; Li & Liang, 2021). To translate that paradigm to our task, we incorporate group membership to the representation learning task by conditioning on the members of the group, assuming the following factorization of the joint distribution over all data:

$$P(Y|X, C) = \prod_{c} \prod_{\{x,y\}\in\mathcal{D}_c} P(y|x, \mathcal{D}_c;\, \theta). \qquad (1)$$

Here, we would concatenate all images belonging to context c to the query image. During ViT processing, these images are patchified and represented as patch tokens, resulting in an input to the Transformer consisting of the tokens belonging to the query image and to the context images. This process is closest to in-context reasoning as commonly applied for LLMs, but ported over to the scenario of representation learning for vision.

In practice, we can subsample patch-tokens from $\mathcal{D}_c$ up to the memory constraints and perform reasoning this way.

2.2. Context-Token Model

Upon further inspection, it is evident that such a process can be interpreted as performing inference over a latent variable explaining away the context-specific covariates t_c as shown in the following:

$$P(y|x, \mathcal{D}_c) = \int P(y|x, t_c;\, \theta) P(t_c|\mathcal{D}_c)\, \mathrm{d}t_c. \qquad (2)$$

Here we assume an implicit hierarchical generative model of images with shared latent structure up to a contextual variable $t_c \in \mathbb{R}^d$ which represents an embedding of the latent covariates determining each distribution and is treated as a *context token*. Such a context token can be characterized by its posterior distribution given the members of the group $P(t_c|\mathcal{D}_c)$ and can then be utilized in a model $P(y|x, t_c)$ to explain the impact of the contextual samples $\mathcal{D}_c$.

Under this viewpoint using Eq.2, we can now see that the context token is shared over all members of a distribution with index c, allowing us to rewrite Eq.1 such that we can break the dependence of the model on $\mathcal{D}_c$ for each data point given this context token:

$$P(Y|X,C) = \prod_c \int P(t_c|\mathcal{D}_c) \prod_{\{x,y\}\in\mathcal{D}_c} P(y|x, t_c;\, \theta)\, \mathrm{d}t_c. \tag{3}$$

We now have established equivalence between the in-context model, and a model which explicitly assumes existence of a context token as additional input to the forward model. We note that recent work proposes a deeply related interpretation of in-context learning in (Xie et al., 2022) supporting our view of an implicit probabilistic model.

2.3. Oracle-Context Model

We now devise simpler formulations of the model and perform maximum likelihood inference over the context token, simplifying Eq. 3 to:

$$P(Y|X,C) = \prod_c \prod_{\{x,y\}\in\mathcal{D}_c} P(y|x, t_c; \theta), \tag{4}$$

where t_c now is a shared parameter that can be inferred per existing group during training. We denote this model the Oracle-Context Model, since it only assumes knowledge of the indicator about which group an image belongs to, and does not require access to the other members of the group.

We can instantiate this oracle model by conditioning the ViTs with a learnable context token t_c and append it to the input sequence of the Transformer: $[t_{\texttt{CLS}}, t_c, t_1 + pos_1, \ldots, t_N + pos_N]$.

The drawback is, that such a method cannot be applied during test-time outside of known distributions, since it will not be able to infer the token embedding without having seen examples from this group during training.

2.4. Context Inference Model

We overcome this limitation by closer matching the objective and assuming a model that performs amortized inference over the context-token parameters on the fly given $\mathcal{D}_c$ by utilizing an inference network $h(\cdot;\phi)$.

Concretely, we will again simplify $p(t_c|\mathcal{D}_c)$ to be given by maximum likelihood but this time also utilize $\mathcal{D}_c$ by posing:

$$P(Y|X,C) = \prod_c \prod_{\{x,y\}\in\mathcal{D}_c} P(y|x;\, t_c, \theta), \quad \text{with } t_c = h(\mathcal{D}_c;\, \phi).$$

Here, $h(\cdot;\phi)$ infers t_c *on the fly* when observing $\mathcal{D}_c$, and can indeed also be used during test time on a previous *unknown distribution* with context c^* time given a set of samples $\mathcal{D}_{c^*}$. There are multiple options available for instantiating the inference model $h(\cdot;\phi)$ (refer to the Appendix for more details). In the case of ContextViT (Figure 1), the inference model $h(\cdot;\phi)$ comprises three key components:

Mean pooling Given an image $x \in \mathcal{D}_c$ with its corresponding patch embeddings t_{p_i}, a direct way to obtain the context token t_c is to aggregate the patch embeddings of all images in $\mathcal{D}_c$ through an average pooling operation

$$t_c^{\text{mean}} = h_c^{\text{mean}}(\mathcal{D}_c) \coloneqq \frac{1}{|\mathcal{D}_c|} \sum_{x\in\mathcal{D}_c} \frac{1}{N} \sum_{x_{p_i}\in x} t_{p_i}.$$

The input sequence for the image x would then be represented as $[t_{\texttt{CLS}}, t_c^{\text{mean}}, t_1 + pos_1, \ldots, t_N + pos_N]$. In practice, instead of pooling over the entire $\mathcal{D}_c$, we apply mean pooling over $\mathcal{B} \cap \mathcal{D}_c$ where $\mathcal{B}$ is the current batch.

Linear transformation with gradients detaching To further enhance the context representation, we introduce a trainable linear transformation to the pooled representation. In order to prevent the patch embeddings from being penalized by the context inference model, we detach their gradients. This results in the expression

$$h_c^{\text{linear}}(\mathcal{D}_c;\, b, W) \coloneqq b + W \cdot \texttt{detach}(t_c^{\text{mean}}).$$

Layerwise context conditioning Recent work (Ghiasi et al., 2022) has shown that Transformer features progress from abstract patterns in early layers to concrete objects in later layers. We explore the application of context conditioning beyond the input layer driven by the hypothesis that patch embeddings may not be able to capture higher-level concepts. For the l-th Transformer layer, we use $y^{(l)}$ to denote its output and $\mathcal{D}_c^{(l)}$ to denote the collection of *hidden* patch embeddings of context c. We propose *layerwise* context conditioning which performs amortized inference over context-token parameters for each layer in the ViT instead of just propagating the input-layer token:

$$P(y^{(l)}|y^{(0)} \coloneqq x; t_c) = \prod_{l=1}^{L} P(y^{(l)}|y^{(l-1)}; t_c^{(l-1)}), \quad \text{with } t_c^{(l)} = h(\mathcal{D}_c^{(l)}; \phi^{(l)}).$$

Table 1. Accuracy of linear probing of pre-trained DINO embeddings on Camelyon17-WILDS. We use † to denote our implementations. Unlabeled ID&OOD denote unlabeled pathology images from all hospitals. In our experiments, we exclude the OOD unlabeled images during the DINO pre-training stage to avoid information leak.

Backbone	Size	Pre-training Method	Pre-training Dataset	In-distribution Accuracy	OOD Accuracy
Fine-tuning all parameters					
DenseNet121 (Koh et al., 2021)	7.9M	Supervised	ImageNet1K	90.6	82.0
DenseNet121 (Sagawa et al., 2021)	7.9M	SwaV	Unlabeled ID&OOD	92.3	91.4
ViT-B/16 (Kumar et al., 2022)	86M	DINO	ImageNet	—	90.6
ViT-L/14 (Kumar et al., 2022)	303M	CLIP	WebImageText	95.2	96.5
Linear probing on frozen backbone					
ViT-L/14 (Kumar et al., 2022)	303M	CLIP	WebImageText	—	92.6
ViT-S/8†	21.7M	DINO	Unlabeled ID	**98.9**	93.8
ContextViT-S/8†	21.8M	DINO	Unlabeled ID	**98.9**	**97.5**

For the l-th Transformer layer, we can express the layerwise context conditioning as

$$[y_{\texttt{CLS}}^{(l)}, y_c^{(l)}, y_{p_1}^{(l)}, \ldots, y_{p_N}^{(l)}] = T^{(l)}([y_{\texttt{CLS}}^{(l-1)}, t_c^{(l-1)}, y_{p_1}^{(l-1)}, \ldots, y_{p_N}^{(l-1)}]).$$

3. Experiments

In medical applications, models are often trained using data from a limited number of hospitals with the intention of deploying them across other hospitals more broadly (Yala et al., 2019; 2021; Bandi et al., 2018). However, this presents a challenge for the generalization of out-of-distribution data. In this section, we aim to evaluate the ability of self-supervised learning with ContextViT to achieve better out-of-distribution generalization. [1]

Dataset We consider the Camelyon17-WILDS benchmark (Bandi et al., 2018; Sagawa et al.). The dataset contains 455,954 labeled and 2,999,307 unlabeled pathology images across five hospitals (3 for training, 1 for validation and 1 for testing). Given a $96 \times 96 \times 3$ image, the task is to predict whether the center region of the image contains any tumor tissue. *We use the hospital id for context inference.*

Evaluation protocol We utilize DINO to pre-train our ViT and ContextViT models from scratch on unlabeled pathology images. Unlike previous work ((Sagawa et al., 2021)) that incorporates both in-distribution and out-of-distribution unlabeled data (Unlabeled ID&OOD), we exclusively use images from the three training hospitals (Unlabeled ID) during pre-training stage to prevent any potential information leakage from out-of-distribution data. Given the 96 by 96 input resolution of our dataset, we opt for a smaller patch size (8 instead of 16) for the ViT models. Further details on our DINO pre-training approach can be found in the Appendix. Once pre-training is complete, we freeze the ViT parameters and train a linear classifier with SGD to predict the target label, the presence of tumors. We use the validation data for model selection and report the testing accuracy on the out-of-distribution hospitals.

Results Table 1 presents a comparison of our linear probing results (marked with†) with other published results on the Camelyon17 benchmark. Our DINO pre-trained ViT-S/8 baseline outperforms the much larger CLIP model in terms of linear probing (93.8 vs. 92.6), which is not surprising given the ViT-S/8 has been pre-trained on the same pathology domain. Next, we see that by conditioning our ViT-S/8 representation with other images from the same hospital within the testing batch, ContextViT-S/8 achieves an OOD accuracy of 97.5%, significantly outperforming all baselines. Due to space constraints, we defer our ablation study and additional analyses to the appendix.

4. Conclusion

We have presented Contextual Vision Transformers, a method that addresses challenges posed by structured variations and covariate shifts in image datasets. By leveraging the concept of in-context learning and introducing context tokens and token inference models, ContextViT enables robust feature representation learning across groups with shared characteristics. Through extensive experimental evaluations across diverse applications, ContextViT consistently demonstrates its utility compared to standard ViT models in terms of out-of-distribution generalization and resilience to batch effects. This success highlights the power and versatility of ContextViT when handling structured variations in real-world applications, where invariance to such nuisances may be desirable.

[1] Given space constraints, we provide our ablation study and analyses in the Appendix. Additionally, we also provide evidence of the utility of ContextViT for supervised fine-tuning on iWildCam and FMoW datasets, as well as for self-supervised representation learning in microscopy imaging on the `cpg-0000` dataset.

References

Assran, M., Caron, M., Misra, I., Bojanowski, P., Bordes, F., Vincent, P., Joulin, A., Rabbat, M., and Ballas, N. Masked siamese networks for label-efficient learning. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXI*, pp. 456–473, 2022.

Bandi, P., Geessink, O., Manson, Q., Van Dijk, M., Balkenhol, M., Hermsen, M., Bejnordi, B. E., Lee, B., Paeng, K., Zhong, A., et al. From detection of individual metastases to classification of lymph node status at the patient level: the camelyon17 challenge. *IEEE transactions on medical imaging*, 38(2):550–560, 2018.

Beery, S., Agarwal, A., Cole, E., and Birodkar, V. The iwildcam 2021 competition dataset. *arXiv preprint arXiv:2105.03494*, 2021.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners, 2020.

Buslaev, A., Iglovikov, V. I., Khvedchenya, E., Parinov, A., Druzhinin, M., and Kalinin, A. A. Albumentations: Fast and flexible image augmentations. *Information*, 11 (2), 2020. ISSN 2078-2489. doi: 10.3390/info11020125. URL https://www.mdpi.com/2078-2489/11/2/125.

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., and Joulin, A. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 9650–9660, 2021.

Chandrasekaran, S. N., Cimini, B. A., Goodale, A., Miller, L., Kost-Alimova, M., Jamali, N., Doench, J., Fritchman, B., Skepner, A., Melanson, M., et al. Three million images and morphological profiles of cells treated with matched chemical and genetic perturbations. *bioRxiv*, pp. 2022–01, 2022.

Christie, G., Fendley, N., Wilson, J., and Mukherjee, R. Functional map of the world. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 6172–6180, 2018.

DeVries, T. and Taylor, G. W. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*.

Gao, I., Sagawa, S., Koh, P. W., Hashimoto, T., and Liang, P. Out-of-distribution robustness via targeted augmentations. In *NeurIPS 2022 Workshop on Distribution Shifts: Connecting Methods and Applications*.

Ghiasi, A., Kazemi, H., Borgnia, E., Reich, S., Shu, M., Goldblum, M., Wilson, A. G., and Goldstein, T. What do vision transformers learn? a visual exploration. *arXiv preprint arXiv:2212.06727*, 2022.

Goyal, S., Kumar, A., Garg, S., Kolter, Z., and Raghunathan, A. Finetune like you pretrain: Improved finetuning of zero-shot vision models. *arXiv preprint arXiv:2212.00638*, 2022.

Haghighi, M., Caicedo, J. C., Cimini, B. A., Carpenter, A. E., and Singh, S. High-dimensional gene expression and morphology profiles of cells across 28,000 genetic and chemical perturbations. *Nature Methods*, pp. 1–8, 2022.

Hao, Y., Sun, Y., Dong, L., Han, Z., Gu, Y., and Wei, F. Structured prompting: Scaling in-context learning to 1,000 examples, 2022.

He, K., Chen, X., Xie, S., Li, Y., Dollár, P., and Girshick, R. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16000–16009, 2022.

Hendrycks, D. and Dietterich, T. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*.

Kaku, A., Mohan, S., Parnandi, A., Schambra, H., and Fernandez-Granda, C. Be like water: Robustness to extraneous variables via adaptive feature normalization. *arXiv preprint arXiv:2002.04019*, 2020.

Khan, S., Naseer, M., Hayat, M., Zamir, S. W., Khan, F. S., and Shah, M. Transformers in vision: A survey. *ACM computing surveys (CSUR)*, 54(10s):1–41, 2022.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Koh, P. W., Sagawa, S., Marklund, H., Xie, S. M., Zhang, M., Balsubramani, A., Hu, W., Yasunaga, M., Phillips, R. L., Gao, I., et al. Wilds: A benchmark of in-the-wild distribution shifts. In *International Conference on Machine Learning*, pp. 5637–5664. PMLR, 2021.

Kumar, A., Shen, R., Bubeck, S., and Gunasekar, S. How to fine-tune vision models with sgd. *arXiv preprint arXiv:2211.09359*, 2022.

Li, X. L. and Liang, P. Prefix-tuning: Optimizing continuous prompts for generation. *arXiv preprint arXiv:2101.00190*, 2021.

Li, Y., Wang, N., Shi, J., Liu, J., and Hou, X. Revisiting batch normalization for practical domain adaptation. *arXiv preprint arXiv:1603.04779*, 2016.

Lin, A. and Lu, A. Incorporating knowledge of plates in batch normalization improves generalization of deep learning for microscopy images. In *Machine Learning in Computational Biology*, pp. 74–93. PMLR, 2022.

Liu, Y., Kothari, P., Van Delft, B., Bellot-Gurlet, B., Mordan, T., and Alahi, A. Ttt++: When does self-supervised test-time training fail or thrive? *Advances in Neural Information Processing Systems*, 34:21808–21820, 2021.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019. URL https://openreview.net/forum?id=Bkg6RiCqY7.

Moshkov, N., Bornholdt, M., Benoit, S., Smith, M., McQuin, C., Goodman, A., Senft, R. A., Han, Y., Babadi, M., Horvath, P., et al. Learning representations for image-based profiling of perturbations. *bioRxiv*, pp. 2022–08, 2022.

Nado, Z., Padhy, S., Sculley, D., D'Amour, A., Lakshminarayanan, B., and Snoek, J. Evaluating prediction-time batch normalization for robustness under covariate shift. *arXiv preprint arXiv:2006.10963*, 2020.

Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Fernandez, P., Haziza, D., Massa, F., El-Nouby, A., et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763. PMLR, 2021.

Sagawa, S., Koh, P. W., Lee, T., Gao, I., Xie, S. M., Shen, K., Kumar, A., Hu, W., Yasunaga, M., Marklund, H., et al. Extending the wilds benchmark for unsupervised adaptation. In *International Conference on Learning Representations*.

Sagawa, S., Koh, P. W., Lee, T., Gao, I., Xie, S. M., Shen, K., Kumar, A., Hu, W., Yasunaga, M., Marklund, H., et al. Extending the wilds benchmark for unsupervised adaptation. *arXiv preprint arXiv:2112.05090*, 2021.

Schneider, S., Rusak, E., Eck, L., Bringmann, O., Brendel, W., and Bethge, M. Improving robustness against common corruptions by covariate shift adaptation. *Advances in Neural Information Processing Systems*, 33: 11539–11551, 2020.

Singh, M., Gustafson, L., Adcock, A., de Freitas Reis, V., Gedik, B., Kosaraju, R. P., Mahajan, D., Girshick, R., Dollár, P., and Van Der Maaten, L. Revisiting weakly supervised pre-training of visual perception models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 804–814, 2022.

Sivanandan, S., Salick, M., Leitmann, B., Liu, K. M., Sultan, M., Ranu, N., Hao, C. V., Chen, O., Bisognano, J., Lubeck, E., et al. Machine learning enabled pooled optical screening in human lung cancer cells. In *NeurIPS 2022 Workshop on Learning Meaningful Representations of Life*.

Sun, Y., Wang, X., Liu, Z., Miller, J., Efros, A., and Hardt, M. Test-time training with self-supervision for generalization under distribution shifts. In *International conference on machine learning*, pp. 9229–9248. PMLR, 2020.

Tompson, J., Goroshin, R., Jain, A., LeCun, Y., and Bregler, C. Efficient object localization using convolutional networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 648–656, 2015.

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.

White, J., Goodman, N., and Hawkins, R. Mixed-effects transformers for hierarchical adaptation, 2022.

Wortsman, M., Ilharco, G., Gadre, S. Y., Roelofs, R., Gontijo-Lopes, R., Morcos, A. S., Namkoong, H., Farhadi, A., Carmon, Y., Kornblith, S., et al. Model soups: averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In *International Conference on Machine Learning*, pp. 23965–23998. PMLR, 2022.

Xie, S. M., Raghunathan, A., Liang, P., and Ma, T. An explanation of in-context learning as implicit bayesian inference, 2022.

Yala, A., Lehman, C., Schuster, T., Portnoi, T., and Barzilay, R. A deep learning mammography-based model for improved breast cancer risk prediction. *Radiology*, 292 (1):60–66, 2019.

Yala, A., Mikhael, P. G., Strand, F., Lin, G., Smith, K., Wan, Y.-L., Lamb, L., Hughes, K., Lehman, C., and Barzilay, R. Toward robust mammography-based models for breast cancer risk. *Science Translational Medicine*, 13(578): eaba4373, 2021.

You, Y., Zhang, Z., Hsieh, C.-J., Demmel, J., and Keutzer, K. Imagenet training in minutes. In *Proceedings of the 47th International Conference on Parallel Processing*, pp. 1–10, 2018.

Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. Deep sets. *Advances in neural information processing systems*, 30, 2017.

Zhang, M., Marklund, H., Dhawan, N., Gupta, A., Levine, S., and Finn, C. Adaptive risk minimization: Learning to adapt to domain shift. *Advances in Neural Information Processing Systems*, 34:23664–23678, 2021.

Zhang, M., Levine, S., and Finn, C. Memo: Test time robustness via adaptation and augmentation. *Advances in Neural Information Processing Systems*, 35:38629–38642, 2022.

Zhou, J., Wei, C., Wang, H., Shen, W., Xie, C., Yuille, A., and Kong, T. Image bert pre-training with online tokenizer. In *International Conference on Learning Representations*.

A. Related Work

Our work shares some ideas and expands on the notion of in-context learning proposed in (Brown et al., 2020) in that we realize that a form of conditioning can position transformer-based models to perform test-time adaptation to new tasks, but instead of conditioning on a dataset explicitly for each query infer context tokens which are shareable across a group. Similar to (Xie et al., 2022) we interpret in-context learning via an implicit generative model, with the key difference that we focus on the case of adaptation to covariate shift and expand that framework with explicit context tokens. Another related recent work builds hierarchical models for transformers applied to language modeling in (White et al., 2022), with similar explaining away of group-specific effects and links to mixed models. Our work differs in the implementation details of this idea from the inference strategy of the context tokens to the application to representation learning to images. Similar to (Hao et al., 2022), our approach also achieves test time adaptation without the enormous memory costs of in-context learning since we share inference over the context token across examples using it and only require a single additional token to represent the group information. Test-time adaptation is also the topic of a wealth of prior work (Sun et al., 2020; Liu et al., 2021; Zhang et al., 2022), while on the fly batch norm statistics have been shown to improve CNN generalization at at test time (Zhang et al., 2021; Nado et al., 2020; Li et al., 2016; Schneider et al., 2020; Lin & Lu, 2022; Kaku et al., 2020). Instead of directly normalizing the feature statistics, our approach abstracts the context information as an extra latent input to the Transformer layer. Our work also demonstrates value in self-supervised representation learning tasks for ViTs (Zhou et al.; He et al., 2022; Assran et al., 2022; Oquab et al., 2023; Caron et al., 2021), yielding robustness to covariate shift through context-conditioning during pre-training.

We also would like to highlight the distinction between previous work on prefix tuning (Li & Liang, 2021) and ContextViT. While both approaches involve conditioning a Transformer model with additional tokens, their objectives and mechanisms differ. Prefix tuning focuses on learning a prefix token for a frozen Transformer model, which can enhance performance for a specific downstream task. In contrast, ContextViT aims to generate in-context feature representations by inferring these tokens directly from the input data. Furthermore, ContextViT jointly learns the Transformer parameters along with the context conditioning, as discussed in Section B.2 and Section B.3.

B. Full experiments

We demonstrate the utilities of ContextViT across a variety of applications. We start with the common supervised fine-tuning setting where we augment three existing pre-trained ViTs with our context inference model (Section B.1). Next we experiment ContextViT with self-supervised representation learning. We demonstrate the importance of context conditioning for both in-distribution generalization B.2 and out-of-distribution generalization B.3. We present our ablation study and analysis in Section B.4.

B.1. Fine-tuning pre-trained ViTs with context conditioning

Dataset To evaluate robustness over data shifts, we consider two image classification datasets from the WILDS benchmark (Koh et al., 2021): iWildCam (Beery et al., 2021) and FMoW (Christie et al., 2018).

iWildCam consists of 203,029 labeled images captured different camera traps. The task is to classify the animal species among 182 possible options. We use the official data split. At test time, we measure the *average classification accuracy* on unseen camera traps to assess out-of-distribution generalization. We use the camera trap id for the context inference.

FMoW includes 141,696 satellite images labeled for 62 different land use categories, such as shopping malls, road bridges, etc. The images are also tagged with the year they were taken and their geographical region. We adopt the standard split, training our model on images from 2002-2013 and validating it on images from 2013-2016. We evaluate the model's *worst-region accuracy* on out-of-distribution images (from 2016-2018) during testing. We use the region id for the context inference.

Evaluation protocol We consider three existing pre-trained ViT models, each with a different number of parameters: 1) DINO ViT-S/8 (Caron et al., 2021), which is pre-trained on ImageNet with self-distillation between a teacher and a student network. 2) SWAG ViT-B/16 (Singh et al., 2022), which is pre-trained on IG3.6B using weak supervision (hashtags from Instagram). 3) CLIP ViT-L/14, which is pre-trained on the Multi-modal WebImageText using language-image contrastive learning. Despite their differences in pre-training objectives and datasets, these models share the same ViT backbone. Therefore we can augment these pre-trained models with our proposed context conditioning mechanism and fine-tune the

Table 2. OOD accuracy on iWIldCam and worst-region OOD accuracy on FMoW for supervised fine-tuning (FT). Augmenting existing pre-trained ViTs (DINO (Caron et al., 2021), SWAG (Singh et al., 2022), CLIP (Radford et al., 2021)) with our context inference model consistently improves the out-of-distribution generalization performance.

	iWildCam			FMoW		
	DINO ViT-S/8	SWAG ViT-B/16	CLIP ViT-L/14	DINO ViT-S/8	SWAG ViT-B/16	CLIP ViT-L/14
FLYP (Goyal et al., 2022)	—	—	$76.2_{\pm 0.4}$	—	—	—
Model Soups (Wortsman et al., 2022)	—	—	$79.3_{\pm 0.3}$	—	—	$47.6_{\pm 0.3}$
FT ViT (Wortsman et al., 2022; Kumar et al., 2022)	—	—	$78.3_{\pm 1.1}$	—	—	49.9
Our implementations						
FT ViT	$75.7_{\pm 0.2}$	$77.5_{\pm 0.5}$	$81.5_{\pm 0.2}$	$35.0_{\pm 0.2}$	$39.8_{\pm 0.5}$	$46.6_{\pm 1.1}$
FT ContextViT (w/o layerwise)	$77.5_{\pm 0.2}$	$78.6_{\pm 0.4}$	$81.9_{\pm 0.1}$	$37.3_{\pm 0.6}$	$41.3_{\pm 1.0}$	$49.1_{\pm 0.7}$
FT ContextViT	$\mathbf{77.7}_{\pm 0.3}$	$\mathbf{79.6}_{\pm 0.6}$	$\mathbf{82.9}_{\pm 0.3}$	$\mathbf{37.7}_{\pm 0.5}$	$\mathbf{41.4}_{\pm 0.3}$	$\mathbf{49.9}_{\pm 0.4}$

combined ContextViT jointly with empirical risk minimization. During fine-tuning, we use standard data augmentation techniques including cropping and resizing, flipping and color jittering. For optimization, we use the AdamW optimizer with a cosine learning rate decay, and we warm up the optimizer for the first 2 epochs. We fine-tune each model for a maximum of 20 epochs with a batch size of 256. We perform model selection based on the accuracy on the official validation split and report the mean and standard deviation across five runs.

Results Table 2 presents our results for supervised fine-tuning. We note that our direct fine-tuning baseline (using CLIP ViT-L/14) outperforms the number reported by (Wortsman et al., 2022) on iWildCam (81.5 vs. 78.3) and underperforms the number reported by (Kumar et al., 2022) on FMoW (46.6 vs. 49.9). We think this is likely caused by the differences in the implementation details (data augmentation, learning rate scheduling, etc.), and unfortunately we cannot find the configuration online to reproduce the exact numbers. Nevertheless, upon comparing our implementations, we make the following observations: 1) Smaller ViTs exhibit inferior generalization compared to larger ViTs; 2) Incorporating our context inference model consistently enhances the performance of ViTs; 3) Layerwise context inference further enhances generalization.

B.2. In-distribution generalization of ContextViT with self-supervised learning

Microscopy imaging with cell painting has demonstrated its effectiveness in studying the effects of cellular perturbations (Haghighi et al., 2022; Moshkov et al., 2022; Sivanandan et al.). Despite meticulous regulation of experimental parameters like cell density and exposure time, technical artifacts can still confound measurements from these screens across different batches. Learning a robust representation for cell images that can effectively handle batch variations remains an ongoing challenge.

Dataset We consider three cell plates (”BR00116991”, ”BR00116993”, ”BR00117000”) from the `cpg0000` dataset released by the JUMP-Cell Painting Consortium (Chandrasekaran et al., 2022). Each plate consists of 384 wells with perturbations (either via a chemical compound or a crispr guide) targeted at 160 different genes. Our input data consists of single-cell images: 229228 images for `BR00116991`, 226311 images for `BR00116993` and 239347 images for `BR00117000`. We note that here each single-cell image has dimension $224 \times 224 \times 8$ (5 fluorescence channels and 3 brightfield channels). For each plate, we split the images randomly into training (40%), validation (10%) and testing (50%). We use the plate id for context inference.

Evaluation protocol For each model, we first use DINO (Caron et al., 2021) to pre-train the ViT and ContextViT from scratch for 100 epochs. Once pre-training is completed, we freeze the ViT parameters and attach a classifier (a MLP with one hidden ReLU layer of dimension 384) to predict the targeted gene of the cell. For each plate, we train the classifier independently using its own training split and report the accuracy on the testing split.

Results Analyzing the first section of Table 3, we note that the representation learned by DINO on a single plate (`BR00116991`) exhibits poor generalization across plates (42.7 for `BR00116993` and 46.8 for `BR00117000`). More-

Table 3. Accuracy of 160-way gene perturbation classification for three cell plates in `cpg-0000`. For each plate, we train a classifier (one-layer MLP) on top of the pre-trained DINO embeddings and evaluate its held out performance. The DINO embeddings are pre-trained on either a single plate `BR00116991` (first section) or all three plates combined (second section).

Method	`BR00116991`	`BR00116993`	`BR00117000`
DINO pre-training on BR00116991 only			
ViT-S/16	56.5	42.7	46.8
DINO pre-training on BR00116991 & BR00116993 & BR00117000			
ViT-S/16	53.6	43.5	45.6
ContextViT-S/16	**57.7**	**48.0**	**48.8**

over, directly combining all three plates for pre-training results in a degradation of the model's performance: -2.9 for `BR00116991`, +0.8 for `BR00116993` and -1.2 for `BR00117000`. In contrast, by utilizing ContextViT, the model effectively accounts for batch effects through context conditioning during the pre-training stage, resulting in superior performance across all three plates.

B.3. Out-of-distribution generalization of ContextViT with self-supervised learning

In medical applications, models are often trained using data from a limited number of hospitals with the intention of deploying them across other hospitals more broadly (Yala et al., 2019; 2021; Bandi et al., 2018). However, this presents a challenge for the generalization of out-of-distribution data. In this section, we aim to evaluate the ability of self-supervised learning with ContextViT to achieve better out-of-distribution generalization.

Dataset We consider the Camelyon17-WILDS benchmark (Bandi et al., 2018; Sagawa et al.). The dataset contains 455,954 labeled and 2,999,307 unlabeled pathology images across five hospitals (3 for training, 1 for validation and 1 for testing). Given a $96 \times 96 \times 3$ image, the task is to predict whether the center region of the image contains any tumor tissue. *We use the hospital id for context inference.*

Evaluation protocol We utilize DINO to pre-train our ViT and ContextViT models from scratch on unlabeled pathology images. Unlike previous work ((Sagawa et al., 2021)) that incorporates both in-distribution and out-of-distribution unlabeled data (Unlabeled ID&OOD), we exclusively use images from the three training hospitals (Unlabeled ID) during pre-training stage to prevent any potential information leakage from out-of-distribution data. Given the 96 by 96 input resolution of our dataset, we opt for a smaller patch size (8 instead of 16) for the ViT models. Once pre-training is complete, we freeze the ViT parameters and train a linear classifier with SGD to predict the target label, the presence of tumors. We use the validation data for model selection and report the testing accuracy on the out-of-distribution hospitals.

Results Table 4 presents a comparison of our linear probing results (marked with†) with other published results on the Camelyon17 benchmark. Our DINO pre-trained ViT-S/8 baseline outperforms the much larger CLIP model in terms of linear probing (93.8 vs. 92.6), which is not surprising given the ViT-S/8 has been pre-trained on the same pathology domain. Next, we see that by conditioning our ViT-S/8 representation with other images from the same hospital within the testing batch, ContextViT-S/8 achieves an OOD accuracy of 97.5%, significantly outperforming all baselines. Moreover, Figure 2 demonstrates that the linear classifier built upon ContextViT-S/8 continues to enhance its OOD performance as the training of the linear classifier progresses, while the one built upon ViT-S/8 exhibits signs of over-fitting to the training data.

B.4. Ablation study and analyses

Context inference model Our ablation study of our context inference model $h(\cdot;\phi)$ is outlined in Table 5. We observe that utilizing the mean patch embeddings as context (94.1) and incorporating an additional linear transformation (94.4) result in improved performance compared to the no context baseline (93.8). Notably, we find that detaching the gradients when calculating the context token from the patch embeddings is crucial, leading to a performance boost from 94.4 to 97.2. Furthermore, the application of layerwise context conditioning further enhance the performance.

We also experimented with other set representation models for context inference, including deep sets (Zaheer et al., 2017). Although deep sets (with detached patch embeddings) outperformed the baseline (94.9 vs. 93.8), it still fell short of the

Table 4. Accuracy of linear probing of pre-trained DINO embeddings on Camelyon17-WILDS. We use † to denote our implementations. Unlabeled ID&OOD denote unlabeled pathology images from all hospitals. In our experiments, we exclude the OOD unlabeled images during the DINO pre-training stage to avoid information leak.

Backbone	Size	Pre-training Method	Pre-training Dataset	In-distribution Accuracy	OOD Accuracy
Fine-tuning all parameters					
DenseNet121 (Koh et al., 2021)	7.9M	Supervised	ImageNet1K	90.6	82.0
DenseNet121 (Sagawa et al., 2021)	7.9M	SwaV	Unlabeled ID&OOD	92.3	91.4
ViT-B/16 (Kumar et al., 2022)	86M	DINO	ImageNet	—	90.6
ViT-L/14 (Kumar et al., 2022)	303M	CLIP	WebImageText	95.2	96.5
Linear probing on frozen backbone					
ViT-L/14 (Kumar et al., 2022)	303M	CLIP	WebImageText	—	92.6
ViT-S/8†	21.7M	DINO	Unlabeled ID	**98.9**	93.8
ContextViT-S/8†	21.8M	DINO	Unlabeled ID	**98.9**	**97.5**

Table 5. Ablation study and pre-training time cost (on a server with 8 A100 GPUs) of different context inference models. For each method, we pre-train the corresponding ContextViT on Camelyon17-WILDS (Unlabeled ID) from scratch using DINO for 100 epochs. We report the out-of-distribution accuracy of linear probing on the official OOD testing split.

Context Inference Method	Linear Probing OOD Acc.	DINO Pre-training Time Cost
No context (ViT-S/8 baseline)	93.8	**21.5h**
Mean patch embeddings	94.1	22.6h
Mean patch embeddings + linear	94.4	23.5h
Mean detached patch embeddings + linear	97.2	23.2h
Layerwise mean detached patch embeddings + linear	**97.5**	29.9h
Deep sets over patch embeddings	92.5	30.0h
Deep sets over detached patch embeddings	94.9	29.1h

simpler approach of mean pooling with linear transformation. Our hypothesis is that the Transformer block's self-attention mechanism, which utilizes the same key, query, and value transformations across the sequence, results in the mean pooling with linear transformation producing a context embedding that is more comparable to the original patch sequence.

Time efficiency One limitation of ContextViT is the extra time cost for the context conditioning. In Table 5, we present the DINO pre-training time cost for different context inference methods. We observe that adding the non-layerwise context conditioning increases the baseline DINO pre-training time by 5%-9%. Applying layerwise context conditioning further increases the time cost by 29%.

Sensitivity to testing batch size During the inference phase, ContextViT-S/8 takes a random batch of testing examples (512 in all previous experiments) and groups them based on their group membership, such as the hospital ID in the case of Camelyon17, to infer the corresponding context tokens. In Figure 3, we present a visualization of ContextViT's linear probing performance sensitivity across different testing batch sizes. Remarkably, even with a testing batch size of 1, where ContextViT-S/8 leverages patches from *a single testing image* (144 patches) to establish its context, it still outperforms our ViT-S/8 baseline by a significant margin (97.0 vs. 93.8). It's worth noting that after DINO pre-training, ContextViT-S/8 acquires the ability to condition its output features with respect to the context. As highlighted by (Gao et al.), pathology images from different hospitals exhibit distinct color distributions. The exceptional performance of ContextViT-S/8 with a small testing batch size demonstrates the model's capability to automatically estimate the color distribution shift from a few representative images during test time.

Attention maps from multiple heads In line with the findings of (Caron et al., 2021), which demonstrate that DINO can learn class-specific features leading to unsupervised object segmentations, we visualize the attention heatmaps for different heads learned with ContextViT in the Camelyon17 dataset. Figure 4 illustrates these attention maps, showcasing that the

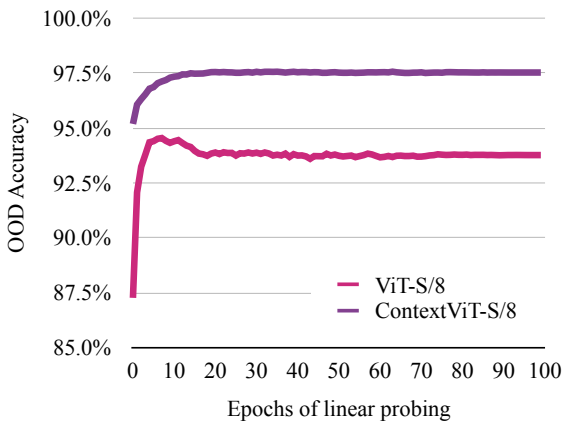

Figure 2. Learning curves of linear probing on Camelyon17. Unlike ViT-S/8, the OOD accuracy on top of ContextViT-S/8 steadily improves as the training of the linear classifier progresses.

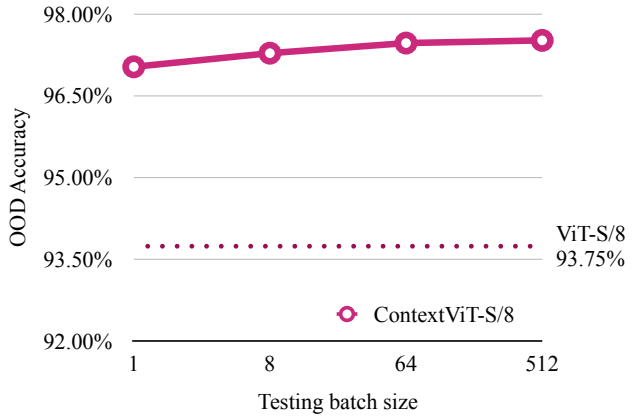

Figure 3. On Camelyon17, we observe that even when we use a testing batch size of 1 (computing the context by aggregating 144 8 x 8 patches from a single testing image into a single context token), ContextViT-S/8 still significantly outperforms ViT-S/8.

learned attentions are focused on meaningful aspects for both in-distribution data and out-of-distribution data. Furthermore, different attention heads exhibit distinct preferences for different cell types. For instance, head-2 primarily focuses on the contour of the cells, while head-3 directs its attention to the actual cells themselves.

Visualizing the context tokens To visualize the learned context tokens for each hospital, we employ PCA and present the results in Figure 5. Remarkably, the inferred context tokens for each hospital exhibit high similarity, appearing closely clustered together. Additionally, we include example images for each hospital on the right side of Figure 5. Notably, the distances between context tokens from different hospitals align well with their visual dissimilarities. For instance, hospital 3 (highlighted in red) is closer to hospital 1 (highlighted in orange) than it is to hospital 4 (highlighted in purple).

C. ContextViT: implementation details

The integration of ContextViT with existing pre-trained ViT models is straightforward. As illustrated in Figure 6, we provide a PyTorch-style pseudo code to outline the process.

C.1. Supervised Fine-tuning

Data augmentation We apply standard data augmentations during supervised fine-tuning for both ContextViTs and ViTs. These augmentations include random resized cropping, random horizontal flipping, and random color jittering.

For the iWildcam dataset, we set the crop scale to be [0.08, 1.0], the horizontal flip probability is set to 0.5, and color jitter is applied with a maximum brightness change of 0.4, maximum contrast change of 0.4, maximum saturation change of 0.4, and maximum hue change of 0.1, with a probability of 0.8.

In the FMoW dataset, objects typically only appear in a small part of the image, so aggressive cropping may introduce noise. Therefore, we set the crop scale to be [0.8, 1.0] and reduce the probability of color jittering to 0.1.

Optimization We conduct fine-tuning of both ViTs and ContextViTs with a batch size of 256 for a total of 20 epochs. For optimization, we follow previous work (Wortsman et al., 2022) and utilize the AdamW optimizer (Loshchilov & Hutter, 2019). To ensure a stable training process, we perform a two-epoch warm-up for the optimizer before applying a cosine annealing scheduler to decay the learning rate. We tune the learning rate within the range of $[10^{-4}, 10^{-5}, 10^{-6}]$. Weight decay is applied to the model, excluding the bias parameters. Inspired by (Caron et al., 2021), we set the weight decay to 0.04 at the beginning and gradually increase it using a cosine scheduler to reach 0.4 at the end of training.

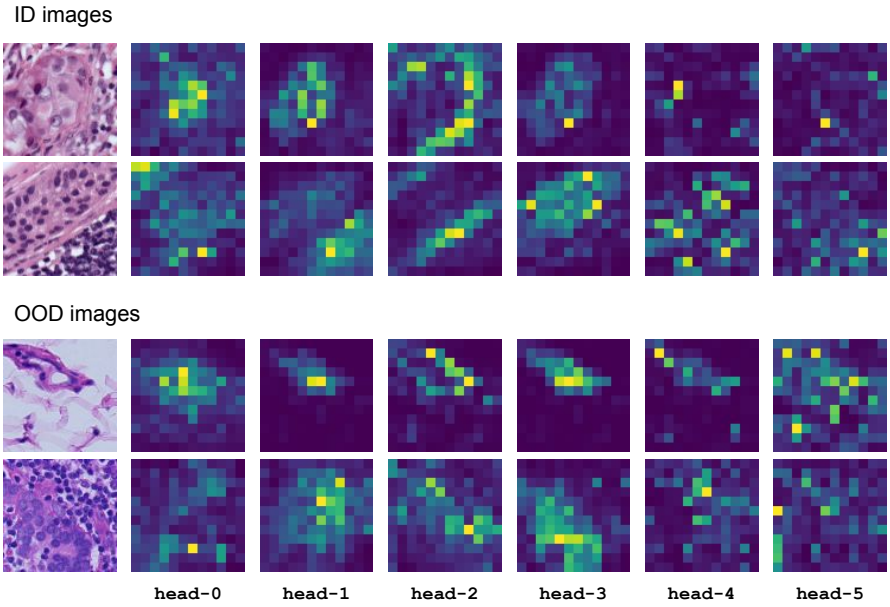

Figure 4. Attention heatmap of ContextViT on patholog images from in-distribution hospitals (top) and OOD hospitals (bottom) in Camelyon17.

C.2. ContextViT on Microscopy Imaging (`cpg-0000`)

DINO pre-training: data augmentation Each single-cell image in the `cpg-0000` dataset has dimensions of 224 by 224 by 8. Since the images are consistently captured at a fixed magnification ratio, we have refrained from applying random scaling, as it could hinder the model's ability to infer the absolute size of the cells accurately. Instead, we have employed the following data augmentations: random padded cropping, horizontal and vertical flipping, rotation (with angles of 90, 180, and 270 degrees), defocus (Hendrycks & Dietterich), coarse dropout (DeVries & Taylor, 2017), and input channel dropout (Tompson et al., 2015). To facilitate these augmentations, we have utilized the Albumentations package (Buslaev et al., 2020), which supports an arbitrary number of channels.

During DINO pre-training, the teacher network receives two global views of the image, while the student network receives six local views. Here, we provide an explanation of the data augmentation configurations for both the global and local views.

For the teacher network in DINO, the two global views are generated as follows: starting with the original single-cell image, we first apply random padding to expand the image to 256×256 and subsequently crop it to 224×224. We apply flipping and rotating transformations uniformly at random. In one view, we apply defocus with a radius range of $(1, 3)$, while in the other view, the defocus radius range is $(1, 5)$. Additionally, we apply coarse dropout, allowing for a maximum of 10 cutout holes, where each hole has a maximum dimension of 10 by 10. However, we do not apply input channel dropout for the global views.

For the student network, which receives eight local views, we follow a similar process as with the global views. Starting with the original image, we apply random padding to expand it to 256×256 and then crop it to 96×96. We apply flipping and rotating transformations uniformly at random, and we use the same defocus radius range of $(1, 3)$ for all six local views. Instead of coarse dropout, we randomly dropout the input channels with a probability of 0.2.

DINO pre-training: optimization Our optimization configuration for the DINO pre-training stage closely follows the guidelines provided in the (Caron et al., 2021) GitHub repository. We utilize a batch size of 512 and train DINO for a total of 100 epochs. The key aspects of our configuration are as follows:

- We employ the AdamW optimizer (Loshchilov & Hutter, 2019) and initiate the learning rate warm-up phase for the first 10 epochs. Considering our batch size, we set the maximum learning rate to 0.001, following recommendations from (You et al., 2018; Caron et al., 2021). Subsequently, we decay the learning rate using a cosine learning rate scheduler.

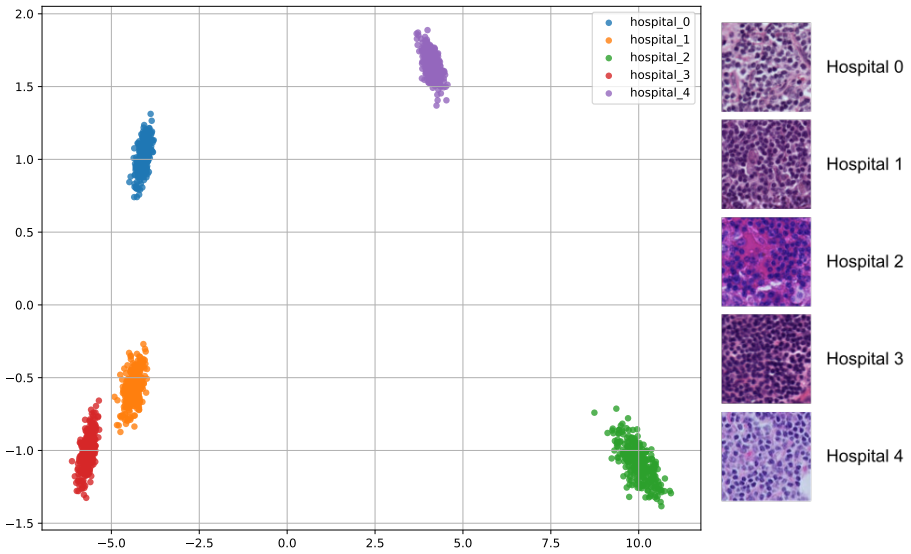

Figure 5. Left: PCA visualization of context tokens inferred by ContextViT using pathology images from different hospitals. Right: Example images from different hospitals.

Our target learning rate at the end of optimization is set to 10^{-6}.

- Weight decay is applied to all parameters except for the biases. We set the initial weight decay to 0.04 and gradually increase it to 0.4 using a cosine learning rate scheduler towards the end of training.
- The DINO projection head we utilize has 65536 dimensions, and we do not employ batch normalization in the projection head.
- The output temperature of the teacher network is initially set to 0.04 and is linearly increased to 0.07 within the first 30 epochs. Throughout the remainder of training, the temperature is maintained at 0.07. Additionally, during the first epoch, we freeze the parameters of the output layer to enhance training stability.

Downstream classifier Based on our preliminary study on `BR00116991` using ViT-S/16, we found that a multi-layer perceptron (MLP) with two hidden layers outperforms the linear classifier (53.6% accuracy vs. 10.4% accuracy). Our final MLP architecture consists of two hidden layers with ReLU activations, and each hidden layer has a dimension of 512. To optimize the parameters of the MLP, we employ the Adam optimizer (Kingma & Ba, 2014) with a batch size of 256, and we train the model for a maximum of 100 epochs. A weight decay of 10^{-5} is applied, and we fine-tune the learning rate within the range of $\in [10^{-3}, 10^{-4}, 10^{-5}]$ to find the optimal value.

C.3. ContextViT on Pathology (Camelyon17-WILDS)

DINO pre-training: data augmentation The Camelyon17-WILDS dataset is a patch-based variant of the Camelyon17 dataset, where each pathology image has a dimension of 96 by 96. To enable finer-grained reasoning in ViTs, we use a smaller patch size of 8 by 8. The images are stored in the RGB pixel format, consisting of 3 bytes per pixel. For pre-training the DINO embeddings, we apply standard data augmentations, as done in previous work (Caron et al., 2021). These augmentations include random resizing and cropping, horizontal flipping, random color jittering, random grayscale transformation, Gaussian blur, and solarization.

Similar to the `cpg-0000` dataset, the teacher network receives two global views of the image, while the student network receives six local views. Here, we provide an explanation of the data augmentation configurations for both the global and local views.

For the teacher network, the two global views are generated as follows: starting with the original pathology image, we first randomly crop the image with a scale sampled from the range of $[0.4, 1.0]$. Then, we resize the cropped image back to its original size of 96 by 96 using a bicubic transformation. A random horizontal flipping is applied with a probability of 0.5.

```
# self.cls_token            The CLS token [1, 1, embed_dim]
# self.pos_embed            The positional embeddings [1, seq_len, embed_dim]
# self.transformer_layers   Sequence (nn.Sequential) of transformer layers.
# self.linear_layers        List (nn.ModuleList) of linear layers.

def forward(self, x, c):
    # Forward pass of ContextViT
    # Arguments:
    #   x    images of dimension: [bs, n_channels, H, W]
    #   c    covariates of x with dimension: [bs]
    # Output:
    #   y    image embeddings: [bs, embed_dim]

    # Convert input images into sequences of patches
    x = self.patch_embed(x)  # [bs, seq_len, embed_dim]

    # Apply the context inference model to compute the context token
    context_token = self.context_inference(x, c, layer_id=0)
    # [bs, 1, embed_dim]

    # concatenate the CLS token, context token and the patch tokens
    x = torch.cat([self.cls_token, context_token, x+self.pos_embed], dim=1)
    # [bs, seq_len+2, embed_dim]

    for layer_id, transformer in enumerate(self.transformer_layers):
        # Apply one transformer layer
        x = transformer(x)  # [bs, seq_len+2, embed_dim]

        # Compute the context token from the hidden patch embeddings
        x[:,1:2] = self.context_inference(x[:,2:], c, layer=layer_id+1)

    # return the image embedding at the cls token
    y = self.norm(x)[:, 0]
    return y


def context_inference(self, x, c, layer_id):
    # Context inference model
    # Arguments:
    #   x          sequences of patch embeddings: [bs, seq_len, embed_dim]
    #   c          covariates of x with dimension: [bs]
    #   layer_id   index of the transformer layer: int
    # Output:
    #   context    context token: [bs, 1, embed_dim]

    # Group patches with the same covariate value
    unique, inverse = torch.unique(c, return_inverse=True)

    # Initialize context token
    context = torch.zeros([bs, 1, embed_dim])

    # Infer context token over examples with the same covariate value
    for idx, u in enumerate(unique):
        # Detached mean pooling over patches that have covariate value u
        m = torch.mean(x[inverse == idx].detach(), dim=[0, 1])

        # Apply a linear layer (based on the transformer layer id)
        context[c==u, 0] = self.linear_layers[layer_id](m)

    return context
```

Figure 6. PyTorch-style pseudo code for ContextViT.

Table 6. OOD accuracy on iWildCam and worst-region OOD accuracy on FMoW for supervised fine-tuning (FT). In "VIT in-context learning", we append 256 random patches, sampled from images within the same group, to the input patch sequence.

	iWildCam			FMoW		
	DINO ViT-S/8	SWAG ViT-B/16	CLIP ViT-L/14	DINO ViT-S/8	SWAG ViT-B/16	CLIP ViT-L/14
Our implementations						
FT ViT	$75.7_{\pm 0.2}$	$77.5_{\pm 0.5}$	$81.5_{\pm 0.2}$	$35.0_{\pm 0.2}$	$39.8_{\pm 0.5}$	$46.6_{\pm 1.1}$
FT ViT in-context learning	$76.3_{\pm 0.6}$	$77.6_{\pm 1.1}$	$81.7_{\pm 0.4}$	$35.1_{\pm 0.6}$	$38.9_{\pm 0.7}$	$46.3_{\pm 1.4}$
FT ContextViT (w/o layerwise)	$77.5_{\pm 0.2}$	$78.6_{\pm 0.4}$	$81.9_{\pm 0.1}$	$37.3_{\pm 0.6}$	$41.3_{\pm 1.0}$	$49.1_{\pm 0.7}$
FT ContextViT	$\mathbf{77.7}_{\pm 0.3}$	$\mathbf{79.6}_{\pm 0.6}$	$\mathbf{82.9}_{\pm 0.3}$	$\mathbf{37.7}_{\pm 0.5}$	$\mathbf{41.4}_{\pm 0.3}$	$\mathbf{49.9}_{\pm 0.4}$

Color jittering is performed with maximum brightness change of 0.4, maximum contrast change of 0.4, maximum saturation change of 0.4, and maximum hue change of 0.1, with a probability of 0.8. The image is transformed into grayscale with a probability of 0.2. Gaussian blur is applied using a kernel size of 1.0. Finally, solarization is applied to one of the global views with a threshold of 0.2.

For the student network, which receives six local views, a similar process is followed with the following changes: 1) The random cropping scale is adjusted to $[0.05, 0.4]$, ensuring that the student network observes only local views of the original image. 2) After cropping, the image is rescaled to a lower resolution of 32 by 32 instead of 96 by 96. 3) Gaussian blur is applied using a larger kernel size of 5. 4) Solarization is not applied to the local views.

DINO pre-training: optimization For the Camelyon17 dataset, we employ the same optimization configuration as for the `cpg-0000` dataset during DINO pre-training. This includes training with a batch size of 512 for 100 epochs, utilizing the AdamW optimizer with a cosine learning rate scheduler, and applying weight decay, among other techniques that we have discussed in Section C.2.

Linear probing To evaluate the resulting DINO embeddings, we utilize the standard linear probing test, which involves training a linear classifier while keeping the ViT backbones frozen. Consistent with (Caron et al., 2021), we employ the following data augmentation and optimization methods:

For data augmentation, we first apply random cropping with a scale range of $[0.08, 1.0]$ and subsequently resize the resulting image to a resolution of 96 by 96. Additionally, we incorporate horizontal flipping with a probability of 0.5. Moreover, we apply color jittering with a probability of 0.8, where the maximum allowed changes include a brightness change of 0.4, a contrast change of 0.4, a saturation change of 0.4, and a hue change of 0.1.

In terms of optimization, we train the linear classifier for 100 epochs using a batch size of 512. Since the only trainable parameters are in the final linear layer, we do not employ warmup or weight decay techniques. To optimize the classifier, we employ SGD with a momentum value of 0.9. The learning rate is selected from the range $[0.0005, 0.001, 0.005]$, and we utilize a cosine annealing scheduler to decay the learning rate gradually throughout the training process.

D. Other options for the context inference model

In this section, we provide a detailed discussion of alternative approaches for implementing the context inference model. We explore various strategies and techniques that can be employed to construct an effective and efficient context inference model, taking into consideration different factors such as computational complexity, model architecture, and inference accuracy. By examining these options, we aim to provide insights and guidance for researchers and practitioners seeking to leverage the concept of context inference in their own applications.

D.1. Handling Covariates with Many Distinct Values

Exponential Moving Average (EMA) While mean pooling has demonstrated excellent performance in all of our experiments, it may not be as effective for covariates with a large number of distinct values. To address this limitation, an alternative approach is to employ an exponential moving average of the patch embeddings. The EMA operation can be

expressed as follows:

$$t_c^{\text{mean}} \leftarrow \lambda t_c^{\text{mean}} + (1-\lambda)\frac{1}{|\mathcal{D}_c|}\sum_{x\in\mathcal{D}_c}\frac{1}{N}\sum_{x_{p_i}\in x} t_{p_i}.$$

Here, $\mathcal{D}c$ represents the dataset containing instances of covariate c, N denotes the number of patches within an image, and tp_i signifies the patch embedding. By incorporating the EMA technique, we can improve the performance of mean pooling for covariates with a large number of distinct values.

Context-based data sampler In all of our experimental settings, we have utilized the standard random sampler, which selects examples uniformly at random from the dataset, for simplicity. However, when dealing with covariates that can take a large number of distinct values, an alternative approach to enhance the performance of direct mean pooling is to employ a context-based data sampler. By sampling a batch comprising examples with the same group membership, we can ensure an adequate number of instances for estimating the context token accurately. Adopting a context-based sampler also offers the advantage of eliminating the need for grouping examples during the forward pass, as the sampler guarantees that all samples within a batch belong to the same group. This can potentially lead to further improvements in computational efficiency. It is worth noting that to prevent divergence across different groups, gradient accumulation across batches may be necessary.

D.2. Exploring Alternative Context Inference Models

The context inference model $h(\cdot;\phi)$ in our study serves as a mapping from a set to a vector representation. In the paper, we propose a simple approach that employs mean pooling followed by a linear transformation. However, there are other options available for parameterizing the context inference model. Here, we discuss additional approaches that can be considered.

Sampling patches from other images in the group As discussed in Section 2.1, one direct way to incorporate context conditioning is by appending image patches from other images within the same group into the patch sequence of the current image. However, this would result in an excessively long sequence length. Since self-attention scales quadratically (in terms of memory and compute) with respect to the sequence length, this approach is not feasible. To overcome this limitation, an alternative is to sample *a fixed number of patches* from these images and utilize them as the context. We explored this baseline approach by sampling 256 patches for the context and present its supervised fine-tuning results in Table 6. The results show mixed performance, likely due to the randomness in the context conditioning. It outperforms the ViT baseline on iWildCam but underperforms ViTs on FMoW. Furthermore, even this sampling approach significantly increases memory consumption compared to ContextViT. For instance, conditioning the CLIP model with 196 patches leads to a 49% increase in GPU memory size.

Deep Sets Deep sets (Zaheer et al., 2017) offers a framework for dealing with objective functions defined on sets that are permutation-invariant. In our application, for each patch token embedding t_{p_i}, it utilizes an encoder network $\varphi(t_{p_i})$ to encode it. Then, it aggregates the embeddings of all patches belonging to the same group into a fixed-length vector using either sum pooling or max pooling. Finally, another network ρ processes the aggregated representation to generate the final output.

In Section 2.4, we conducted experiments with a specific instance of the deep sets model, where we employed two multi-layer perceptrons (each with two hidden layers and ReLU activations) as the φ and ρ networks. Additionally, we incorporated residual connections for each hidden layer. We utilized the sum pooling mechanism to aggregate information across the set. As shown in Table 5, while this method exhibits increased representation power compared to mean pooling with a linear transformation, the latter demonstrates better generalization capabilities.

Transformer Another approach for aggregating information across sets is by utilizing the Transformer architecture (Vaswani et al., 2017). The Transformer layer, without the inclusion of positional information, inherently possesses permutation invariance and can be employed as a set encoder. To obtain the final set representation, an additional classifier token can be appended to the input sequence. However, due to the substantial computational requirements associated with this method, we did not explore it in our experiments.