CASS: Cross Architectural Self-Supervision for Medical Image Analysis

Pranav Singh
Department of Computer Science
Tandon School of Engineering
New York University New York, NY 11202
ps4364@nyu.edu

Elena Sizikova
Center for Data Science
New York University New York, NY 10011
es5223@nyu.edu

Jacopo Cirrone
Center for Data Science
New York University
and Colton Center for Autoimmunity
NYU Grossman School of Medicine
New York, NY 10011
cirrone@courant.nyu.edu

Abstract

Recent advances in deep learning and computer vision have reduced many barriers to automated medical image analysis, allowing algorithms to process label-free images and improve performance. However, existing techniques have extreme computational requirements and drop a lot of performance with a reduction in batch size or training epochs. This paper presents Cross Architectural - Self Supervision (CASS), a novel self-supervised learning approach that leverages Transformer and CNN simultaneously. Compared to the existing state of the art self-supervised learning approaches, we empirically show that CASS-trained CNNs and Transformers across four diverse datasets gained an average of 3.8% with 1% labeled data, 5.9% with 10% labeled data, and 10.1% with 100% labeled data while taking 69% less time. We also show that CASS is much more robust to changes in batch size and training epochs. Notably, one of the test datasets comprised histopathology slides of an autoimmune disease, a condition with minimal data that has been underrepresented in medical imaging.

1 Introduction

In recent years, medical image analysis has seen tremendous growth due to the availability of powerful computational modelling tools, such as neural networks, and the advancement of techniques capable of learning from partial annotations. Medical imaging is a field characterized by minimal data availability. First, data labeling typically requires domain-specific knowledge. Therefore, the requirement of large-scale clinical supervision may be cost and time prohibitive. Second, due to patient privacy, disease prevalence, and other limitations, it is often difficult to release imaging datasets for secondary analysis, research, and diagnosis. Third, due to an incomplete understanding of diseases. This could be either because the disease is emerging or because no mechanism is in place to systematically collect data about the prevalence and incidence of the diseases. An example of the latter is autoimmune diseases. Statistically, autoimmune diseases affect 3% of the US population or 9.9 million US citizens. On the other hand, 4% of the US population or 13.6 million US citizens are affected by cancer. Cancer has been widely studied in medical science as well as at the intersection

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

of medical science and artificial intelligence. However,the application of artificial intelligence in medical image analysis for autoimmune diseases has received much less attention. Furthermore, there are still major outstanding research questions for autoimmune diseases regarding the presence of different cell types and their role in inflammation at the tissue level. Their study is critical not only because they affect a large part of society but also because they have been on the rise recently Galeotti and Bayry [2020], Lerner et al. [2015], Ehrenfeld et al. [2020].

To overcome these limitations, we turn to self-supervised learning, a learning paradigm that allows for learning useful data representations label-free. Models extracting these representations can later be fine-tuned with a small amount of labeled data for each downstream task Sriram et al. [2021]. As a result, this learning approach avoids the relatively expensive and human-intensive task of data annotation and makes it an effective tool for the image analysis of emerging diseases that often have limited data availability (e.g., dermatomyositis, an autoimmune disease, or COVID-19, the cause of a recent worldwide pandemic). Existing approaches in the field of self-supervised learning rely on Convolutional Neural Networks (CNNs) or Transformers as the feature extraction backbone and learn feature representations by teaching the network to compare the extracted representations. Instead, we propose to combine a CNN and Transformer in a response-based contrastive method. In CASS, the extracted representations of each input image are compared across two branches representing each architecture (see Figure 1). By transferring features sensitive to translation equivariance and locality from CNN to Transformer, CASS learns more predictive data representations in limited data scenarios where a Transformer-only model cannot find them. We studied this qualitatively and quantitatively in Section 5. Our contributions are as follows:

- We introduce **C**ross **A**rchitectural - **S**elf **S**upervision (CASS), a hybrid CNN-Transformer approach for learning improved data representations in a self-supervised setting in limited data availability problems in the medical image analysis domain [1].
- We evaluate CASS on three challenging medical image analysis problems (autoimmune disease cell classification, brain tumor classification, and skin lesion classification) on four datasets (Dermatomyositis dataset Van Buren et al. [2022], Dermofit Project Dataset Fisher and Rees [2017], brain tumor MRI Dataset Cheng [2017], Kang et al. [2021] and ISIC 2019 Tschandl et al. [2018], Gutman et al. [2018], Combalia et al. [2019]) and find that CASS outperforms existing state of the art self-supervised techniques by an average of 3.8% using 1% label fractions, 5.9 % with 10% label fractions and 10.1% with 100% label fractions.
- Existing methods also suffer a severe drop in performance when trained for a reduced number of epochs or batch size (Caron et al. [2021], Grill et al. [2020], Chen et al. [2020]). We show that CASS is robust to these changes in Section 5.6.
- New state of the art self-supervised techniques often require significant computational requirements. This is a major hurdle as these methods can take around 20 GPU days to train Azizi et al. [2021]. This makes them inaccessible in limited computational resource settings and increase triage in medical image analysis. CASS, on average, takes 69% less time as opposed to the existing state of the art methods. We further expand on this result in Appendix A.

2 Methodology

CASS' goal is to extract and learn representations in a self-supervised way. To achieve this, an image is passed through a common set of augmentations. The augmented image is then simultaneously passed though a CNN and Transformer to create positive pairs. The output logits from the CNN (shown with a R) and Transformer (shown with a T) are then used to find cosine similarity based loss (equation 1). Since, the two architectures give different output representations as mentioned in Raghu et al. [2021], the model doesn't collapse. We also report results for CASS using different set of CNNs and Transformers in Appendix B and C, and not a single case of model collapse was registered. In Figure 1, we show CASS on top and DINO Caron et al. [2021] at the bottom.

[1]The code is open source and available at: github.com/pranavsinghps1/CASS

$$\text{loss} = 2 - 2 \times (\sum_{i=1}^{N} \left(\frac{R}{(\max \|R\|_2), \epsilon)} \right)) \times (\sum_{i=1}^{N} \left(\frac{T}{(\max (\|T\|_2), \epsilon)} \right)) \quad (1)$$

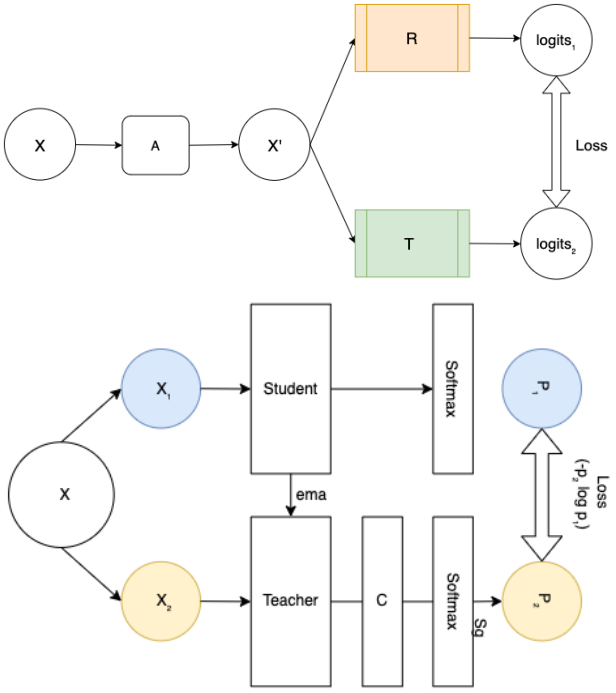

Figure 1: (Top) In our proposed self-supervised architecture - CASS, R represents ResNet-50, a CNN and T in the other box represents the Transformer used (ViT); X is the input image, which becomes X' after applying augmentations. Note that CASS applies only one set of augmentations to create X'. X' is then passed through both the arms to compute loss as mentioned in Equation 1. This is different from DINO, which passes different augmentation of the same image through networks with the same architecture but different parameters. Another key difference is that in CASS, loss is computed over logits meanwhile in DINO it is computed over softmax output.

3 Results

3.1 Quantitative performance

We provide the compressed results over four datasets in Table 1, we further expand upon the results in Appendix A and B. Furthermore, we also expand our study to include ablation studies on change in batch size, training epochs, changing the two architectures used, augmentations, optimization and initialization in Appendix B and C.

3.2 Qualitative performance

To qualitatively understand the gain of CASS trained CNN and Transformer over supervised CNN and Transformers, we compare the attention maps of Transformers and feature maps of CNNs. This would give us an insight into the innate nature of CNNs and Transformer and whats the effect of training both of them simultaneously with CASS. To do this we studied the class attention maps and feature maps of CASS and supervised CNN as well as Transformers over a sample image in Figures 2 and 3. We also expand this to study class attention maps and feature maps over multiple samples over all datasets in Appendix D.

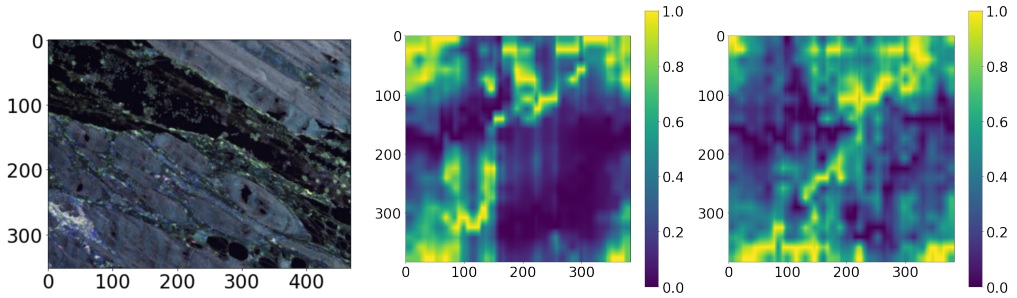

Figure 2: Overall attention maps from supervised Transformer (on the left) and CASS trained Transformer (on the right). We pass the same image as input through both of them; the image used is shown in Figure 2. We observed that the CASS-trained Transformer's attention is more spread as compared to supervised Transformer. This can be easily inferred from the right-hand side bottom portion of both the attention maps.

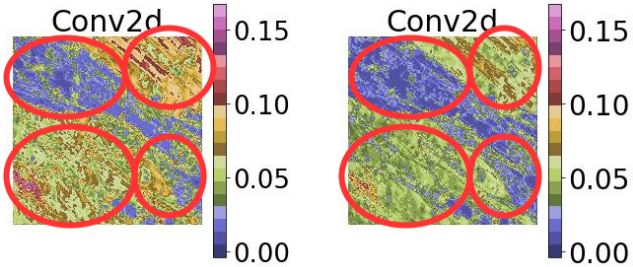

Figure 3: This figure shows the feature map extracted after the first layer of ResNet-50 for CASS (on the left) and supervised CNN (on the right). From the four circles, it is clear that CASS-trained CNN can retain much more intricate detail about the input image (Figure 2) that the supervised CNN misses.

	1% labeled data		10% labeled data		100% labeled data	
	CNN	Transformer	CNN	Transformer	CNN	Transformer
Autoimmune Dataset						
DINO	-	-	**0.8237**	0.844	0.84252	0.8639
CASS	-	-	0.8158	**0.8717**	**0.8650**	**0.8894**
Supervised	-	-	0.82095	0.8356	0.831	0.8420
Dermofit Dataset						
DINO	-	-	0.3749	0.332	0.6775	0.4810
CASS	-	-	**0.4367**	**0.3896**	**0.7132**	**0.6667**
Supervised	-	-	0.33	0.299	0.6341	0.456
Brain MRI Dataset						
DINO	**0.63405**	0.3211	**0.92325**	0.7529	0.9900	0.8841
CASS	0.40816	**0.3345**	0.8925	**0.7833**	**0.9909**	**0.9279**
Supervised	0.52	0.3017	0.9022	0.747	0.9899	0.8719
ISIC-2019 Dataset						
DINO	0.328	0.3676	0.3797	0.3998	0.493	0.5408
CASS	**0.3617**	**0.3973**	**0.41**	**0.4395**	**0.543**	**0.5819**
Supervised	0.2640	0.3074	0.3070	0.3586	0.35	0.42

Table 1: In this table we show the performance of DINO, CASS and supervised training for ResNet-50 as CNN and ViT B/16 as Transformer over our four experimental datasets. We further expand on these results in Appendix A.2.

4 Conclusion

Based on our experimentation on four diverse medical imaging datasets, we qualitatively and empirically conclude that CASS gained an average of 3.8% with 1% labeled data, 5.9% with 10% labeled data, and 10.1% with 100% labeled data and trained in 69% less time than the existing state of the art self-supervised method. Furthermore, we saw that CASS is robust to batch size changes and training epochs reduction. To conclude, for medical image analysis, CASS is computationally efficient, performs better, and overcomes some of the shortcomings of existing self-supervised techniques. This ease of accessibility and better performance will catalyze medical imaging research to help us improve healthcare solutions and develop new solutions for underrepresented and emerging diseases.

References

Caroline Galeotti and Jagadeesh Bayry. Autoimmune and inflammatory diseases following covid-19. *Nature Reviews Rheumatology*, 16(8):413–414, 2020.

Aaron Lerner, Patricia Jeremias, and Torsten Matthias. The world incidence and prevalence of autoimmune diseases is increasing. *Int J Celiac Dis*, 3(4):151–5, 2015.

Michael Ehrenfeld, Angela Tincani, Laura Andreoli, Marco Cattalini, Assaf Greenbaum, Darja Kanduc, Jaume Alijotas-Reig, Vsevolod Zinserling, Natalia Semenova, Howard Amital, et al. Covid-19 and autoimmunity. *Autoimmunity reviews*, 19(8):102597, 2020.

Anuroop Sriram, Matthew Muckley, Koustuv Sinha, Farah Shamout, Joelle Pineau, Krzysztof J Geras, Lea Azour, Yindalon Aphinyanaphongs, Nafissa Yakubova, and William Moore. Covid-19 prognosis via self-supervised representation learning and multi-image prediction. *arXiv preprint arXiv:2101.04909*, 2021.

Kayla Van Buren, Yi Li, Fanghao Zhong, Yuan Ding, Amrutesh Puranik, Cynthia A. Loomis, Narges Razavian, and Timothy B. Niewold. Artificial intelligence and deep learning to map immune cell types in inflamed human tissue. *Journal of Immunological Methods*, 505:113233, 2022. ISSN 0022-1759. doi: https://doi.org/10.1016/j.jim.2022.113233. URL `https://www.sciencedirect.com/science/article/pii/S0022175922000205`.

Robert Fisher and Jonathan Rees. Dermofit project datasets. 2017. https://homepages.inf.ed.ac.uk/rbf/DERMOFIT/datasets.htm.

Jun Cheng. brain tumor dataset. 4 2017. doi: 10.6084/m9.figshare.1512427.v5. URL `https://figshare.com/articles/dataset/brain_tumor_dataset/1512427`.

Jaeyong Kang, Zahid Ullah, and Jeonghwan Gwak. Mri-based brain tumor classification using ensemble of deep features and machine learning classifiers. *Sensors*, 21(6), 2021. ISSN 1424-8220. doi: 10.3390/s21062222. URL `https://www.mdpi.com/1424-8220/21/6/2222`.

Philipp Tschandl, Cliff Rosendahl, and Harald Kittler. The ham10000 dataset, a large collection of multi-source dermatoscopic images of common pigmented skin lesions. *Scientific Data*, 5, 2018.

David A. Gutman, Noel C. F. Codella, M. E. Celebi, Brian Helba, Michael Armando Marchetti, Nabin K. Mishra, and Allan C. Halpern. Skin lesion analysis toward melanoma detection: A challenge at the 2017 international symposium on biomedical imaging (isbi), hosted by the international skin imaging collaboration (isic). *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*, pages 168–172, 2018.

Marc Combalia, Noel C. F. Codella, Veronica M Rotemberg, Brian Helba, Verónica Vilaplana, Ofer Reiter, Allan C. Halpern, Susana Puig, and Josep Malvehy. Bcn20000: Dermoscopic lesions in the wild. *ArXiv*, abs/1908.02288, 2019.

Mathilde Caron, Hugo Touvron, Ishan Misra, Hervé Jégou, Julien Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

Jean-Bastien Grill, Florian Strub, Florent Altch'e, Corentin Tallec, Pierre H. Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Ávila Pires, Zhaohan Daniel Guo, Mohammad Gheshlaghi Azar, Bilal Piot, Koray Kavukcuoglu, Rémi Munos, and Michal Valko. Bootstrap your own latent: A new approach to self-supervised learning. *ArXiv*, abs/2006.07733, 2020.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020.

Shekoofeh Azizi, Basil Mustafa, Fiona Ryan, Zach Beaver, Jana von Freyberg, Jonathan Deaton, Aaron Loh, Alan Karthikesalingam, Simon Kornblith, Ting Chen, Vivek Natarajan, and Mohammad Norouzi. Big self-supervised models advance medical image classification. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 3458–3468, 2021.

Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? In *NeurIPS*, 2021.

Ross Wightman. Pytorch image models. `https://github.com/rwightman/pytorch-image-models`, 2019.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 10012–10022, October 2021.

Maithra Raghu, Chiyuan Zhang, Jon Kleinberg, and Samy Bengio. Transfusion: Understanding transfer learning for medical imaging. *Advances in neural information processing systems*, 32, 2019.

Bill Cassidy, Connah Kendrick, Andrzej Brodzicki, Joanna Jaworek-Korjakowska, and Moi Hoon Yap. Analysis of the isic image datasets: usage, benchmarks and recommendations. *Medical Image Analysis*, 75:102305, 2022.

Nils Gessert, Maximilian Nielsen, Mohsin Shaikh, René Werner, and A. Schlaefer. Skin lesion classification using ensembles of multi-resolution efficientnets with meta data. *MethodsX*, 7, 2020.

Javeria Amin, Muhammad Almas Anjum, Muhammad Sharif, Saima Jabeen, Seifedine Kadry, and Pablo Moreno Ger. A new model for brain tumor detection using ensemble transfer learning and quantum variational classifier. *Computational Intelligence and Neuroscience*, 2022, 2022.

David Picard. Torch. manual_seed (3407) is all you need: On the influence of random seeds in deep learning architectures for computer vision. *arXiv preprint arXiv:2109.08203*, 2021.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

Tsung-Yi Lin, Priya Goyal, Ross B. Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2999–3007, 2017.

A Result Analysis

A.1 Time complexity analysis

From Table 2, we observed that CASS takes 69% less time as compared to DINO. This reduction in time could be attributed to the follwoing reasons:

1. In DINO, augmentations are applied twice as opposed to just once in CASS. Furthermore, we per application CASS uses less number of augmentations as compared to DINO.
2. Since the architectures, used are different, there is no scope of parameter sharing between them. A major chunk of time is saved in just updating the two architectures, after each epoch instead of re-initializing architectures with lagging parameters.

Dataset	DINO	CASS
Autoimmune	1 Hour 13 Mins	**21 Mins**
Dermofit	3 Hours 9 mins	**1 Hour 11 Mins**
Brain MRI	26 Hours 21 Mins	**7 Hours 11 Mins**
ISIC-2019	109 Hours 21 Mins	**29 Hours 58 Mins**

Table 2: Self-supervised training time comparison for 100 epochs on a single RTX8000 GPU. The 100 epoch mark was chosen because, beyond that, the ISIC-2019 dataset was taking more than 168 hours to train, which was more than the ceiling time we could request on our internal cluster. Hence due to computational limitations, we chose 100 epochs as our stopping mark.

A.2 Complete results

In this section we present expanded results of Table 1. We preformed each experiments 5 times and report the average metrics in the following tables.

A.2.1 Autoimmune Diseases Biopsy Slides Dataset

We did not perform 1% training for the autoimmune diseases biopsy slides of 198 images because using 1% images would be too small number to learn anything meaningful and the results would be highly randomized.

Following the self-supervised training and fine tuning procedure as described in Appendix E.2 and E.3, we observed that using CASS with the ViT B/16 backbone and ResNet50 improved upon existing result of 0.63 Van Buren et al. [2022] to 0.8894. All kinds of Transformers consistently outperformed CNNs.

Techniques	Backbone	Testing F1 score	
		10%	100%
DINO	Resnet-50	**0.8237±0.001**	0.84252±0.008
CASS	Resnet-50	0.8158±0.0055	**0.8650±0.0001**
Supervised	Resnet-50	0.82095±0.007	0.831±0.0216
DINO	ViT B/16	0.8445±0.0008	0.8639± 0.002
CASS	ViT B/16	**0.8717±0.005**	**0.8894±0.005**
Supervised	ViT B/16	0.8356±0.007	0.8420±0.009

Table 3: Results for autoimmune biopsy slides dataset. In this table we compare the F1 score on test set. We observed that CASS outperformed the existing state-of-art self-supervised method using 100% labels for CNN as well as for Transformers. Although DINO outperforms CASS for CNN with 10% labeled fraction. Overall CASS outperforms DINO by 2.2% for 100% labeled training for CNN and Transformer. For Transformers in 10% labeled training CASS' performance was 2.7% better than DINO.

A.3 Dermofit Dataset

Similar to the autoimmune dataset, we did not perform 1% training for this dataset as the training set was too small to draw meaningful results with just 10 samples. We observed that CASS outperformed both supervised and existing state of the art self-supervised methods for all label fractions. We present the F1 score for different label fractions in Table 4.

A.4 Brain tumor MRI dataset

We observed that supervised CNNs performed better than Transformers on this dataset. Similarly, for DINO, CNNs performed better than Transformers by a margin. We observed that this trend is followed by CASS as well. The difference between CNN and Transformer performance is smaller for CASS as compared to the difference in performance for CNN and Transformer with supervised and DINO training. We report these results in Table 5.

Techniques	Testing F1 score	
	10%	100%
DINO (Resnet-50)	0.3749±0.0011	0.6775±0.0005
CASS (Resnet-50)	**0.4367±0.0002**	**0.7132±0.0003**
Supervised (Resnet-50)	0.33±0.0001	0.6341±0.0077
DINO (ViT B/16)	0.332± 0.0002	0.4810±0.0012
CASS (ViT B/16)	**0.3896±0.0013**	**0.6667±0.0002**
Supervised (ViT B/16)	0.299±0.002	0.456±0.0077

Table 4: Results for the dermofit dataset. Parenthesis next to the techniques represent the architecture used, for example DINO(ViT B/16) represents ViT B/16 trianed with DINO. In this table we compare the F1 score on test set. We observed that CASS outperformed the existing state-of-art self-supervised method using for all label fractions and for both the architectures.

A.5 ISIC 2019 Dataset

The ISIC-2019 dataset is an incredibly challenging dataset, not only because of the class imbalance issue but because it is made of partially processed and inconsistent images with hard-to-classify classes. From Table 6 it is clear that CASS outperforms DINO for all label fractions for both CNN and Transformer by a margin.

B Ablation Study

B.1 Batch size

As mentioned, with CASS, we aim to overcome the shortcomings of existing self-supervised methods where they drop much performance with a reduction in batch size. To study this effect, we ran CASS with three different batch sizes on the autoimmune dataset - 8, 16, and 32 and reported the results in 7. We observed that the performance of CASS-trained CNN improved with a reduction in batch size while that of the Transformer remained almost constant. We followed the standard set of protocols mentioned in Appendix E for training. As standard, we used 16 as our batch size. We also conducted this experiment on the brain MRI classification dataset and reported the results in Table 15.

B.2 Change in training epochs

As standard, we trained CASS for 100 epochs in all cases. However, existing self-supervised techniques are plagued with a loss in performance with a decrease in the number of training epochs. To test this effect for CASS, we ran it for 50, 100, 200, and 300 epochs on the autoimmune dataset and reported the results in Table 8. We observed a slight gain in performance when we increased the epochs from 100 to 200 but minimal gain beyond that. We also ran the same experiment on the brain MRI dataset and reported the results in Table 16.

Techniques	Backbone	Testing F1 score		
		1%	10%	100%
DINO	Resnet-50	**0.63405±0.09**	**0.92325±0.02819**	0.9900±0.0058
CASS	Resnet-50	0.40816±0.13	0.8925±0.0254	**0.9909± 0.0032**
Supervised	Resnet-50	0.52±0.018	0.9022±0.011	0.9899± 0.003
DINO	ViT B/16	0.3211±0.071	0.7529±0.044	0.8841± 0.0052
CASS	ViT B/16	**0.3345±0.11**	**0.7833±0.0259**	**0.9279± 0.0213**
Supervised	ViT B/16	0.3017 ± 0.077	0.747±0.0245	0.8719± 0.017

Table 5: While DINO outperformed CASS for 1% and 10% labeled training for CNN, CASS maintained its superiority for 100% labeled training, albeit by just 0.09%. Similarly, CASS outperformed DINO for all data regimes for Transformers, incrementally 1.34% in for 1%, 3.04% for 10%, and 4.38% for 100% labeled training. We observe that this margin is more significant than for biopsy images. Such results could be ascribed to the increase in dataset size and increasing learnable information.

Techniques	Backbone	Testing Balanced multi-class accuracy		
		1%	10%	100%
DINO	Resnet-50	0.328±0.0016	0.3797±0.0027	0.493±3.9e-05
CASS	Resnet-50	**0.3617±0.0047**	**0.41±0.0019**	**0.543±2.85e-05**
Supervised	Resnet-50	0.2640±0.031	0.3070±0.0121	0.35±0.006
DINO	ViT B/16	0.3676± 0.012	0.3998±0.056	0.5408±0.001
CASS	ViT B/16	**0.3973± 0.0465**	**0.4395±0.0179**	**0.5819±0.0015**
Supervised	ViT B/16	0.3074±0.0005	0.3586±0.0314	0.42±0.007

Table 6: Results for the ISIC-2019 dataset. Comparable to the official metrics used in the challenge `https://challenge.isic-archive.com/landing/2019/`. We use balanced multi-class accuracy as our metric, which is semantically equal to recall value. We observed that CASS consistently outperforms DINO by approximately 4% for all label fractions with CNN and Transformer.

Batch Size	CNN F1 Score	Transformer F1 Score
8	0.88285±0011	0.8844±0.0009
16	0.8650±0.0001	0.8894±0.005
32	0.8648±0.0005	0.889±0.0064

Table 7: F1 metric comparison between the two arms of CASS trained over 100 epochs, following the protocols and procedure listed in Appendix E. We only change the batch size during self-supervised training. Based on these we observed that while CASS trained Transformer gains 1% with reduction in batch size from 16 to 8, CASS trained CNN gains almost 2% for the same change. Although there is a diminishing gain in performance as we increase the batch size.

Epochs	CNN F1 Score	Transformer F1 Score
50	0.8521±0.0007	0.8765± 0.0021
100	0.8650±0.0001	0.8894±0.005
200	0.8766±0.001	0.9053±0.008
300	0.8777±0.004	0.9091±8.2e-5

Table 8: Performance comparison over varied number of epochs, from 50 to 300 epochs, the downstream training procedure and the CNN-Transformer combination is kept constant across all the four experiments, only the number of self-supervised epochs have been changed.

B.3 Augmentations

Contrastive learning techniques are known to be highly dependent on augmentations. Recently, most self-supervised techniques have adopted BYOL Grill et al. [2020]-like a set of augmentations. DINO Caron et al. [2021] uses the same set of augmentations as BYOL, along with adding local-global cropping. We use a reduced set of BYOL augmentations for CASS, along with a few changes. For instance, we do not use solarize and Gaussian blur. Instead, we use affine transformations and random perspectives. In this section, we study the effect of adding BYOL-like augmentations to CASS. We report these results in Table 9. We observed CASS trained CNN is robust to changes in augmentations. On the other hand, the Transformer drops performance with change in augmentations. A possible solution to regain this loss in performance for Transformer with change in augmentation is by using Gaussian blur, which has a converging effect on the results of CNN and Transformer.

Augmentation Set	CNN F1 Score	Transformer F1 Score
CASS only	0.8650±0.0001	0.8894±0.005
CASS + Solarize	0.8551±0.0004	0.81455±0.002
CASS + Gaussian blur	0.864±4.2e-05	0.8604±0.0029
CASS + Gaussian blur + Solarize	0.8573±2.59e-05	0.8513±0.0066

Table 9: We report the F1 metric of CASS trained with a different set of augmentations for 100 epochs. While CASS-trained CNN fluctuates within a percent of its peak performance, CASS-trained Transformer drops performance with the addition of solarization and Gaussian blur. Interestingly, the two arms converged with the use of Gaussian blur.

B.4 Optimization

In CASS we use Adam optimizer for both CNN and Transformer. This is a shift from the traditional use of SGD or stochastic gradient descent for CNNs. In this Table 10 we report the performance of CASS trained CNN and Transformer with the CNN using SGD and Adam optimizer. We observed that while the performance of CNN remained almost constant, the performance of Transformer dropped by almost 6% with CNN using SGD.

Optimiser for CNN	CNN F1 Score	Transformer F1 Score
Adam	0.8650±0.0001	0.8894±0.005
SGD	0.8648±0.0005	0.82355±0.0064

Table 10: We report the F1 metric of CASS trained with a different set of optimizers for the CNN arm for 100 epochs. While there is no change in CNN's performance, the Transformer's performance drops around 6% with SGD.

B.5 Change in architecture

B.5.1 Changing Transformer and keeping the CNN same

From Table 11 and 12, we observed that CASS trained ViT Transformer with the same CNN consistently gained approximately 4.7% over its supervised counterpart. Furthermore, from Table 12 we observed that although ViT L/16 performs better than ViT B/16 on ImageNet (Wightman [2019]'s results), we observed that the trend is opposite on the autoimmune dataset. Hence, the supervised performance of architecture must be considered before pairing it with CASS on the specific datatset.

Transformer	CNN F1 Score	Transformer F1 Score
ViT Base/16	0.8650±0.001	0.8894± 0.005
ViT Large/16	0.8481±0.001	0.853±0.004

Table 11: In this table we show performance of CASS for ViT large/16 with ResNet-50 and ViT base/16 with ResNet-50. We observed that CASS trained Transformers on average performed 4.7% better than their supervised counterparts.

Architecture	Testing F1 Score
ResnNet-50	0.831±0.0216
ViT Base/16	0.8420±0.009
ViT large/16	0.80495±0.0077

Table 12: Supervised performance of ViT family on the autoimmune dataset. We observed that as opposed to ImageNet performance, ViT large/16 performs worse than ViT Base/16 on the autoimmune dataset.

B.5.2 Changing CNN and keeping the Transformer same

Table 13 and 14 we observed that similar to changing Transformer while keeping CNN same, CASS trained CNNs gained an average of 3% over their supervised counterparts. For ResNet-200 Wightman [2019] doesn't have ImageNet initialization hence used random initialization.

CNN	Transformer	100% Label Fraction	
		CNN F1 score	Transformer F1 score
ResNet-18 (**11.69M**)		0.8674±4.8e-5	0.8773±5.29e-5
ResNet-50 (**25.56M**)	ViT Base/16 (**86.86M**)	0.8680±0.001	0.8894± 0.0005
ResNet-200 (**64.69M**)		0.8517±0.0009	0.874±0.0006

Table 13: F1 metric comparison between the two arms of CASS trained over 100 epochs, following the protocols and procedure listed in Appendix E. The numbers in parentheses show the parameters learned by the network. We use Wightman [2019] implementation of CNN and transformers, with ImageNet initialisation except for ResNet-200.

Architecture	Testing F1 Score
ResnNet-18	0.8299±0.0004
ResnNet-50	0.831±0.0216
ResnNet-200	0.823±0.0005

Table 14: Supervised performance of the ResNet CNN family on the autoimmune dataset.

C Additional Results

We conducted some further experimentation to check for collapse and ablation studies. The model did not report collapse in any case. The following results are calculated following the protocols in Appendix E on the brain MRI classification dataset.

C.1 Changing Batch Size

This section presents the results of varying the batch size in the brain MRI classification dataset. In the standard implementation of CASS, we used a batch size of 16; here, we show results for batch sizes 8 and 32. The largest batch size we could run was 34 on a single GPU of 48 GB video memory. Hence 32 was the biggest batch size we showed in our results. We present these results in Table 15. However, the performance of CNN remained nearly constant, with a change of less than a percentage. The performance of the Transformer drops as we increase the batch size. Since CASS was developed with the requirements of running on small datasets, with an overall size smaller than the batch size of the current state of the art techniques, its peak performance for small batch size justifies its development. Furthermore, from Table 7 and Section A.2.1, we observed that Transformer is the better performing architecture out of the two and that with batch size change, it was unaffected. Similarly, from Table 15 and Section A.4, we observed that CNN is the better performing architecture, and with batch size change, its performance is unaffected. Hence, we conclude that batch size change does not affect the leading architecture of a given dataset.

Batch Size	CNN F1 Score	Transformer F1 Score
8	0.9895±0.0025	0.93158±0.0109
16	0.9909± 0.0032	0.9279± 0.0213
32	0.9848±0.011	0.9176±0.006

Table 15: This table represents the results for different batch sizes on the brain MRI classification dataset. We maintain the downstream batch size constant for all the three self-supervised batch sizes, following the standard experimental setup as mentioned in Appendix E. These results are for 100% label fraction.

C.2 Effect of the Number of Training Epochs

We saw that there was an incremental gain in performance as we increased the number of self-supervised training epochs. For the opposite scenario, there wasn't a steep drop in performance like the existing self-supervised techniques when we reduce the number of self-supervised epochs. Table 16 displays results for this experimentation.

Epochs	CNN F1 Score	Transformer F1 Score
50	0.9795±0.0109	0.9262±0.0181
100	0.9909± 0.0032	0.9279± 0.0213
200	0.9864±0.008	0.9476±0.0012
300	0.9920±0.001	0.9484±0.017

Table 16: Performance comparison over varied number of epochs, from 50 to 300 epochs, the downstream training procedure and the CNN-transformer combination is kept constant across all the four experiments, only the number of self-supervised epochs has been changed.

C.3 Effect of Changing the ViT/CNN branch

C.3.1 Changing CNN while keeping Transformer same

For this experiment we use ResNet family of CNNs along with ViT base/16 as our Transformer. We use ImageNet initialization for ResNet 18 and 50, while random initialization for ResNet-200. We present these results in Table 17. We observed that increase in performance of ResNet correlates to increase in performance of Transformer, hence implying that there is information transfer between the two.

CNN	Transformer	100% Label Fraction	
		CNN F1 score	Transformer F1 score
ResNet-18 (**11.69M**)	ViT Base/16 (**86.86M**)	0.9913±0.002	0.9801±0.007
ResNet-50 (**25.56M**)		0.9909±0.0032	0.9279± 0.0213
ResNet-200 (**64.69M**)		0.9898±0.005	0.9276±0.017

Table 17: F1 metric comparison between the two arms of CASS trained over 100 epochs, following the protocols and procedure listed in Appendix E. The numbers in parentheses show the parameters learned by the network. We use Wightman [2019] implementation of CNN and transformers, with ImageNet initialisation except for ResNet-200.

C.3.2 Changing Transformer while keeping CNN same

For this experiment we keep the CNN as constant and study the effect of changing the Transformer. For this experiment we use ResNet as our choice of CNN and ViT base and large Transformers with 16 patches. Additionally we also report performance for DeiT-B with ResNet-50. We report these results in Table 18. Similar to Table 11 we observe that changing Transformer from ViT Base to Large while keeping the number of tokens same at 16, performance drops. Additionally, for approximately the same size, out of DEiT base and ViT base Trasnformers, DEiT performs much better than ViT base.

C.3.3 Using CNN in both arms

Until now we have experimented by using a CNN and a Transformer in CASS. In this section we present results for using two CNNs in CASS. We pair ResNet-50 with DenseNet-161. We observe that both the CNNs fail to reach the benchmark set by ResNet-50 and ViT-B/16 combination. Although training the ResNet-50-DenseNet-161 pair takes 5 hour 24 minutes which is less than the 7 hours 11 minutes taken by the ResNet-50-ViT-B/16 combination to be trained with CASS. We compare these results in Table 19.

C.3.4 Using Transformer in both arms

Similar, to the above section, for this section, we use a Transformer-Transformer combination instead of a CNN-Transformer combination. For this we use Swin-Transformer patch-4/window-12 Liu et al. [2021] alongside ViT-B/16 Transformer. We observe that the performance for ViT/B-16 improves by around 1.3% when we use Swin Transformer. However, this comes at a computational cost. Swin-ViT combination took 10 hours to train as opposed to 7 hours 11 minutes taken by the ResNet-50-ViT-B/16 combination to be trained with CASS. Even with the increased time to train the Swin-ViT combination, it is still almost 50% less than DINO. We present these results in Table 20.

C.4 Effect of Initialization

ImageNet initialization is preferred for transfer learning in medical image analysis, not because of feature reuse but because ImageNet weights allow for faster convergence through better weight scaling Raghu et al. [2019]. We use ImageNet initialized CNN and Transformers for CASS, DINO, and supervised training. We use Timm's library for this initialization Wightman [2019]. However, sometimes pretrained weights might be hard to find, so we study CASS' performance with random and ImageNet initialization in this section. We observed that performance almost remained the same with minor gains when the initialization was altered for the two networks. Table 21 presents the results for this experimentation.

CNN	Transformer	CNN F1 Score	Transformer F1 Score
ResNet-50 (25.56M)	DEiT Base/16 (86.86M)	0.9902±0.0025	0.9844±0.0048
	ViT Base/16 (86.86M)	0.9909±0.0032	0.9279± 0.0213
	ViT Large/16 (304.72M)	0.98945±2.45e-5	0.8896±0.0009

Table 18: For the same number of Transformer parameters, DEiT-base with ResNet-50 performed much better than ResNet-50 with ViT-base. The difference in their CNN arm is 0.10%. On ImageNet DEiT-base has a top1% accuracy of 83.106 while ViT-base has an accuracy of 86.006. We use both the Transformers with 16 patches. [ResNet-50 has an accuracy of 80.374]

CNN	Architecture in arm 2	F1 Score of ResNet-50 arm	F1 Score of arm 2
ResNet-50	ViT Base/16	0.9909±0.0032	0.9279± 0.0213
	DenseNet-161	0.9743±8.8e-5	0.98365±9.63e-5

Table 19: We observed that for the ResNet-50-DenseNet-161 pair, we train 2 CNNs instead of 1 in our standard setup of CASS. Furthermore, none of these CNNs could match the performance of ResNet-50 trained with the ResNet-50-ViT base/16 combination. Hence, by adding a Transformer-CNN combination, we transfer information between the two architectures that would have been missed otherwise.

Architecture in arm 1	Transfomer	F1 Score of arm 1	F1 Score of ViT-B/16 arm
ResNet-50	ViT Base/16	0.9909±0.0032	0.9279± 0.0213
Swin Tranformer		0.9883±1.26e-5	0.94±8.12e-5

Table 20: We present the results for using Transformers in both the arms and compare the results with CNN-Transformer combination.

Initialisation	CNN F1 Score	Transformer F1 Score
Random	0.9907±0.009	0.9316±0.027
Imagenet	0.9909±0.0032	0.9279± 0.0213

Table 21: We observe that the Transformer gains some performance with random initialization, although performance has more variance when used with random initialization.

D Qualitative Results

We reported expanded quantitative results and ablation studies in Appendix A, B and C respectively. In this section, we study the qualitative gains of the cross architectural self-supervised training using a CNN and Transformer simultaneously. We start by studying the feature maps of the first few layers of CASS and supervised ResNet-50 on the autoimmune dataset. We also studied the class attention maps of CASS and supervised ViT base/16 Transformers over the four datasets.

D.1 Feature maps

In this section, we study the feature maps from the first five layers of the ResNet-50 model trained with CASS and supervision. We extracted feature maps after the Conv2d layer of ResNet-50. We present the extracted features in Figure 5. We observed that CASS-trained CNN could retain much more detail about the input image than supervised CNN.

D.2 Class attention maps

We have already studied the class attention maps over a single image in Section 3. This section will study the average class attention maps for all four datasets. We studied the attention maps averaged over 30 random samples for autoimmune, dermofit, and brain MRI datasets. Since the ISIC 2019 dataset is highly unbalanced, we averaged the attention maps over 100 samples so that each class may have an example in our sample. We maintained the same distribution as the test set, which has the same class distribution as the overall training set. We observed that CASS-trained Trasformers

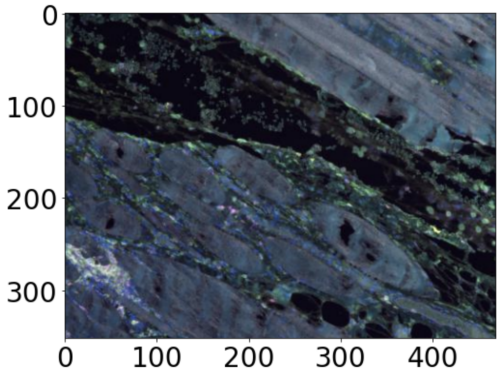

Figure 4: Sample image used from the test set of the autoimmune dataset.

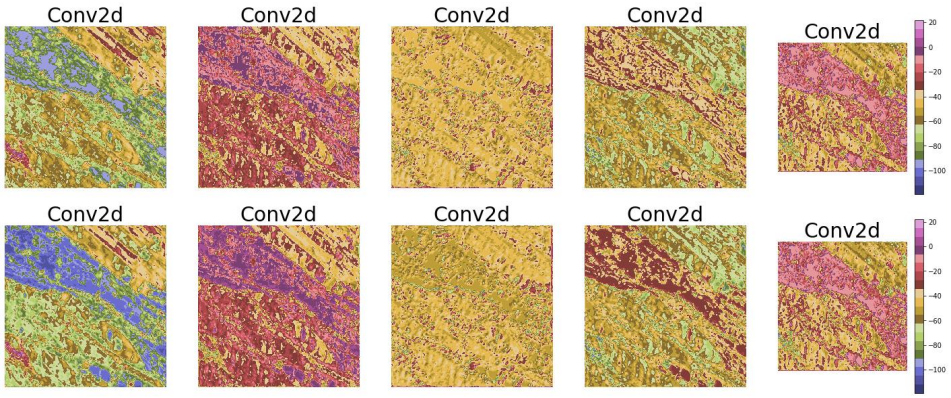

Figure 5: At the top we have features extracted from top 5 layers of supervised ResNet-50 while at the bottom we have features extracted from top 5 layers of CASS trained ResNet-50. We supplied both the networks with the same input (shown in Figure 4).

were better able to map global and local attention maps due to Transformers ability to map global dependencies and by learning features sensitive to translation equivariance and locality from CNN.

D.2.1 Autoimmune dataset

We study the class attention maps averaged over 30 test samples for the autoimmune dataset in Figure 6. We observed that the CASS-trained Transformer has much more attention in the center as compared to the supervised Transformer. This extra attention could be attributed that a Transformer on its own couldn't map out is due to the information transfer from CNN. Another, feature to observe is that the attention map of CASS-trained Transformer is much more connected than that of supervised Transformer.

D.2.2 Dermofit dataset

We present the average attention maps for the dermofit dataset in Figure 7. We observed that the CASS-trained Transformer is able to pay a lot more attention to the center part of the image. Furthermore, the attention map of CASS-trained Transformer is much more connected as compared to the supervised Transformer. So, overall with CASS, the Transformer is not only able to map long-range dependencies which are innate to Transformers but is also able to make more local connections with the help of features sensitive to translation equivariance and locality from CNN.

D.2.3 Brain tumor MRI classification dataset

We present the results for the average class attention maps in Figure 8. We observed that a CASS-trained Transformer could better capture long and short-range dependencies than a supervised Transformer. Furthermore, we observed that a CASS-trained Transformer's attention map is much more centered than a supervised Transformer's. From Figure 12 we can observe that most MRI images are center localized, so having a more centered attention map is advantageous in this case.

D.2.4 ISIC 2019 dataset

The ISIC-2019 dataset is one of the most challenging datasets out of the four datasets. ISIC 2019 consists of images from the HAM10000 and BCN_20000 datasets Cassidy et al. [2022], Gessert et al. [2020]. For the HAM1000 dataset, it is difficult to classify between 4 classes (melanoma and melanocytic nevus), (actinic keratosis and benign keratosis). HAM10000 dataset contains images of size 600×450, centered and cropped around the lesion. Histogram corrections have been applied to only a few images. The BCN_20000 dataset contains images of size 1024×1024. This dataset is particularly challenging as many images are uncropped, and lesions are in difficult and uncommon locations. Hence, in this case, having more spread-out attention maps would be advantageous instead of a more centered one. From Figure 9, we observed that CASS-trained Transformer has a lot more spread attention map than a supervised Transformer. Furthermore, CASS-trained Transformer is also able to attend the corners far better than supervised Transformer.

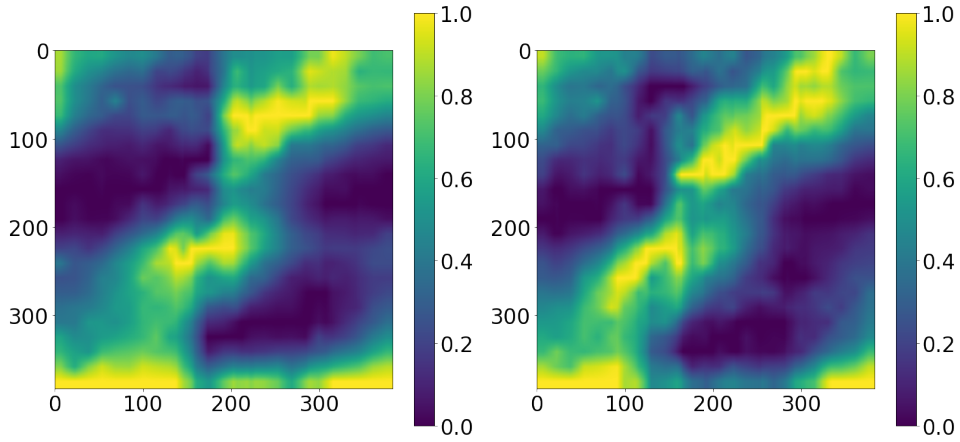

Figure 6: To ensure the consistency of our study, we studied average attention maps over 30 sample images from the autoimmune dataset. The top image is the overall attention map averaged over 30 samples for supervised Transformer, while the one at the bottom is for CASS trained Transformer.

E Expansion on experimentation details

E.1 Datasets

E.1.1 Choice of Datasets

We chose four medical imaging datasets with diverse sample sizes ranging from 198 to 25,336 and diverse modalities to study the performance of existing self-supervised techniques and CASS. Most of the existing self-supervised techniques have been studied on million image datasets, but medical imaging datasets, on average, are much smaller than a million images. We expand this to include datasets of emerging and underrepresented diseases with only a few hundred samples, like the autoimmune dataset in our case (198 samples). To the best of our knowledge, no existing literature studies the effect of self-supervised learning on such a small dataset. Furthermore, we chose the dermofit dataset because all the images are taken using an SLR camera, and no two images are the same size. Image size in dermofit varies from 205×205 to 1020×1020. MRI images constitute a large part of medical imaging; hence we included this dataset in our study. So, to incorporate them in our study, we included the Brain tumor MRI classification dataset. Furthermore, it is our

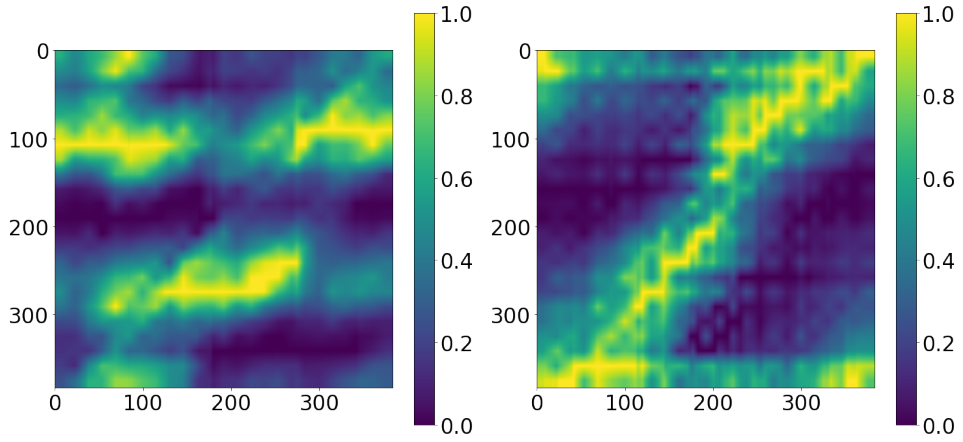

Figure 7: Class attention maps averaged over 30 samples of the dermofit dataset for supervised Transformer (on the left), and CASS trained Transformer (on the right).

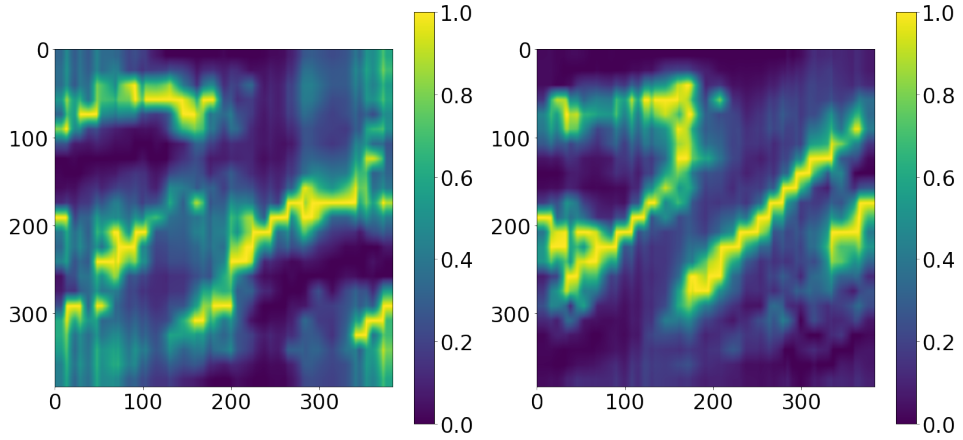

Figure 8: Class attention maps averaged over 30 samples of the brain tumor MRI classification dataset for supervised Transformer (on the left) and CASS trained Transformer (on the right).

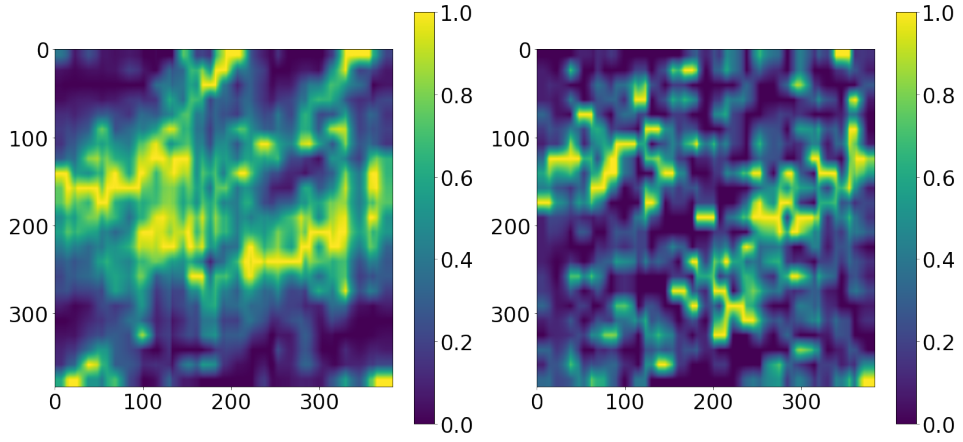

Figure 9: Class attention maps avergaed over 100 samples form the ISIC-2019 dataset for supervised Transformer (on the left) and CASS trained Transformer (on the right).

study's only black and white dataset; the other three datasets are RGB. The ISIC 2019 is a unique dataset as it contains multiple pairs of hard-to-classify classes (Melanoma - melanocytic nevus and actinic keratosis - benign keratosis) and different image sizes - out of which only a few have been prepossessed. It is a highly imbalanced dataset containing samples with lesions in difficult and uncommon locations. We further expand on the datasets in the following section.

- **Autoimmune diseases biopsy slides** (Van Buren et al. [2022]) consists of slides cut from muscle biopsies of dermatomyositis patients stained with different proteins and imaged to generate a dataset of 198 TIFF image set from 7 patients. The presence or absence of these cells helps to diagnose dermatomyositis. Multiple cell classes can be present per image; therefore this is a multi-label classification problem. Our task here was to classify cells based on their protein staining into TFH-1, TFH-217, TFH-Like, B cells, and others. We used F1 score as our metric for evaluation, as employed in the original work by Van Buren et al. [2022]. These RGB images have a consistent size of 352 by 469.

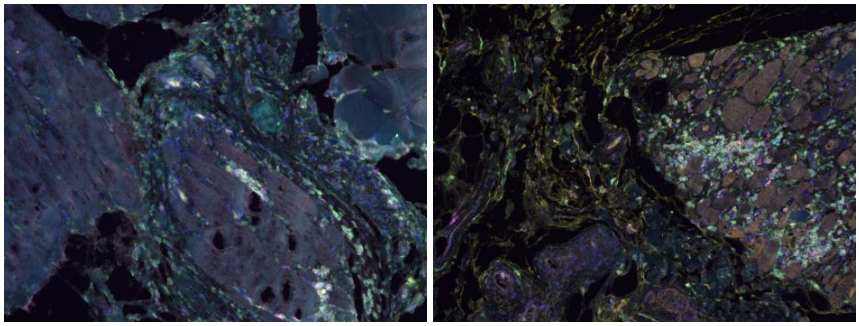

Figure 10: Sample of autofluorescence slide images from the muscle biopsy of patients with dermatomyositis - a type of autoimmune disease.

- **Dermofit dataset** Fisher and Rees [2017] contains normal RGB images captured through SLR camera indoors with ring lightning. There are 1300 image samples, classified into 10 classes: Actinic Keratosis (AK), Basal Cell Carcinoma (BCC), Melanocytic Nevus / Mole (ML), Squamous Cell Carcinoma (SCC), Seborrhoeic Keratosis (SK), Intraepithelial carcinoma (IEC), Pyogenic Granuloma (PYO), Haemangioma (VASC), Dermatofibroma (DF) and Melanoma (MEL). This dataset comprises of images of different sizes and no two images are of same size. They range from 205×205 to 1020×1020 in size. We pretext task is multi-class classification and we use F1 score as our evaluation metric on this dataset.

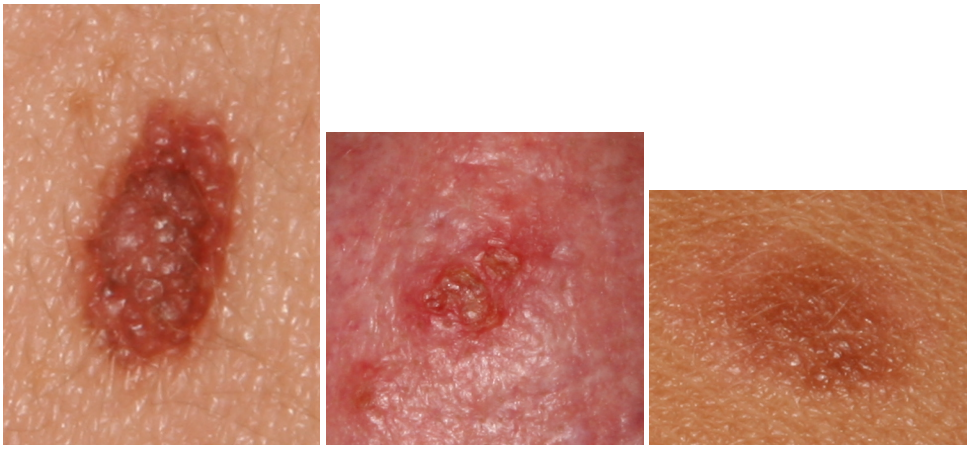

Figure 11: Sample images from the Dermofit dataset.

- **Brain tumor MRI dataset** Cheng [2017], Amin et al. [2022] 7022 images of human brain MRI that are classified into four classes: glioma, meningioma, no tumor, and pituitary.

We used the dataset from `https://www.kaggle.com/datasets/masoudnickparvar/brain-tumor-mri-dataset` that combines Br35H: Brain tumor Detection 2020 dataset used in "Retrieval of Brain tumors by Adaptive Spatial Pooling and Fisher Vector Representation" and Brain tumor classification curated by Navoneel Chakrabarty and Swati Kanchan. Out of these, the dataset curator created the training and testing splits. We followed their splits, 5,712 images for training and 1,310 for testing. Since this was a combination of multiple datasets, size of images vary throughout the dataset from 512×512 to 219×234. The pretext of the task is multi-class classification, and we used the F1 score as the metric.

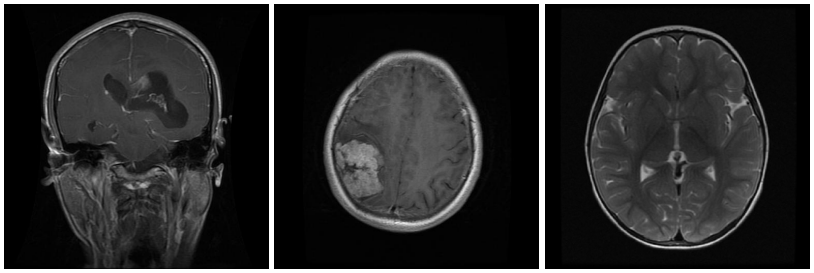

Figure 12: Sample images of brain tumor MRI dataset, Each image corresponds to a prediction class in the data set glioma (Left), meningioma (Center) and No tumor (Right)

- **ISIC 2019** (Tschandl et al. [2018], Gutman et al. [2018], Combalia et al. [2019]) consists of 25,331 images across eight different categories - melanoma (MEL), melanocytic nevus (NV), Basal cell carcinoma (BCC), actinic keratosis(AK), benign keratosis(BKL) , dermatofibroma(DF), vascular lesion (VASC) and Squamous cell carcinoma(SCC). This dataset contains images of size 600×450 and 1024×1024. The distribution of these labels is unbalanced across different classes. For evaluation, we followed the metric followed in the official competition i.e balanced multi-class accuracy value, which is semantically equal to recall.

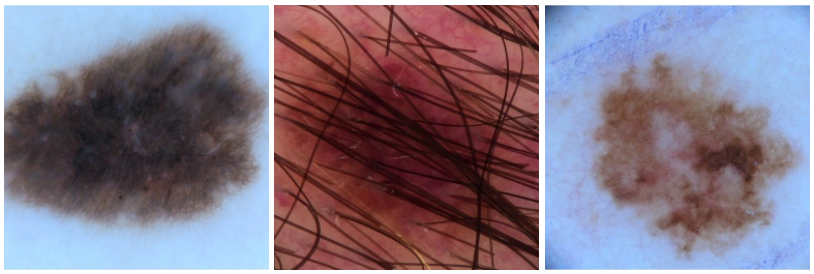

Figure 13: Sample images from the ISIC-2019 challenge dataset.

E.2 Self-supervised training

E.2.1 Protocols

- Self-supervised learning was only done on the training data and not on the validation data. We used `https://github.com/PyTorchLightning/pytorch-lightning` to set the pseudo-random number generators in PyTorch, NumPy and (python.random).
- We run training over five different seed values, and report mean results with variance in each table. We don't perform a seed value sweep to extract anymore performance Picard [2021].
- For DINO implementation we use Phil Wang's implementation: `https://github.com/lucidrains/vit-pytorch`.
- For implementation of CNNs and Transformers we use timm's library Wightman [2019].
- For all experiments, ImageNet Deng et al. [2009] initialised CNN and Transformers were used.

E.2.2 Augmentations

- Resizing: Resize input images to 384×384 with bilinear interpolation.
- Color jittering: change the brightness, contrast, saturation and hue of an image or apply random perspective with a given probability. We set the degree of distortion to 0.2 (between 0 and 1) and use bilinear interpolation, with an application probability of 0.3.
- Color jittering or apply random affine transformation of the image keeping center invariant with degree 10, with an application probability of 0.3.
- Horizontal and Vertical flip. Each with an application probability of 0.3.
- Channel normalisation with mean (0.485, 0.456, 0.406) and standard deviation (0.229, 0.224, 0.225).

E.2.3 Hyper-parameters

- Optimization: We use stochastic weighted averaging over Adam optimiser with learning rate (LR) set to 1e-3 for both CNN and vision transformer (ViT). This is a shift from SGD which is usally used for CNNs.
- Learning Rate: Cosine annealing learning rate is used with 16 iterations and a minimum learning rate of 1e-6. Unless mentioned otherwise, this setup was trained over 100 epochs. These were then used as initialisation for the downstream supervised learning. The standard batch size is 16.

E.3 Supervised training

E.3.1 Augmentations

We use the same set of augmentations used in self-supervised training.

E.3.2 Hyper-parameters

- We use Adam optimiser with lr set to 3e-4 and a cosine annealing learning schedule.
- Since, all medical datasets have class imbalance we address it by using focal loss Lin et al. [2017] as our choice of loss function with the alpha value set to 1 and the gamma value to 2. In our case it uses minimum-maximum normalised class distribution as class weights for focal loss.
- We train for 50 epochs. We also use a five epoch patience on validation loss to check for early stopping. This downstream supervised learning setup is kept the same for CNN and Transformers.

We repeated all the experiments five time with different seed values and then present the average results in all the tables (except Table 1).

E.4 Self-supervised Training

We studied and compared results between DINO and CASS trained self-supervised CNNs and Transformers. For the same, we trained from ImageNet initialization for 100 epochs with a batch size of 16. All experiments were conducted on an internal cluster with single GPU unit (NVIDIA RTX8000) with 48 GB video RAM, 2 CPU cores and 64 GB system RAM.

For DINO, we used the hyper parameters and augmentations mentioned in the original implementation. For CASS, we describe the experimentation details in Appendix E.2.

After self-supervised training perform end-to-end fine tuning on each dataset, we expand on this in the next section.

E.5 End-to-end fine-tuning

In order to evaluate the utility of the learned representations, we use the self-supervised pre-trained weights for downstream classification task. While performing the downstream fine tuning we perform

entire model (E2E fine-tuning). The test set metrics were used as proxies for representation quality. We trained the entire model for a maximum of 50 epochs with an early stopping patience of 5 epochs. For supervised fine tuning we used Adam optimizer with a cosine annealing learning rate starting at 3e-04. Since almost all medical datasets have some class imbalance we applied class distribution normalized Focal Loss Lin et al. [2017] to navigate class imbalance. We fine tune the models with different label fractions during training i.e 1%, 10% and 100% label fractions after self-supervised pre-training. For example, if a model is trained with 10% label fraction then that model will have access only to 10% of the training dataset labels fine tuning, which was done after self-supervised training.

F Self-supervised Algorithm

The core self-supervised algorithm, used to train CASS with a CNN (R) and a Transformer (T), is described in Algorithm 1. Here, num_epochs represents the number of self-supervised epochs to run. CNN and Transformer represents the respective architecture we use, for example CNN could be a ResNet50 and Transformer can be ViT Base/16. Finally, after training using the algorithm described in Algorithm 1, we save the CNN and Transformer for downstream E2E finetuning.

Algorithm 1: CASS self-supervised training algorithm

Input: Unlabeled same augmented images from the training set x'
Output: Logits from each network.
Data: Images from a given dataset

for *epochs in range(num_epochs)* **do**
  for *x in train loader:* **do**
    $R = cnn(x')$ `// taking logits output from CNN`
    $T = vit(x')$ `// taking logits output from ViT`
    Calculate loss using Equation 1`// taking cosine similarity between the logits outputs from CNN and ViT`
    Calculate the mean value of all elements of the loss tensor. Compute gradients.