Master Thesis

Self-Supervised Transformer Model via Hierarchical-Masked-Attention for Medical Image Analysis

Simon Philipp Baur

Supervisors: Dr. Mina Rezaei
Date: February 24^{th}, 2023

Declaration of Originality

I confirm that the submitted thesis is original work and was written by me without further assistance. Appropriate credit has been given where reference has been made to the work of others.

Munich, February 24^{th}, 2023

. . . . . .
Simon Philipp Baur

Abstract

In this work we propose Hierarchical-Masked-Attention, a stochastic extension of the Self-Attention-Mechanism in Vision Transformer Models, that masks a number of attention heads during training and thins out masking over training time, according to a newly introduced Hyper-Parameter h_p. We apply our method in a Self-Supervised Learning framework (DINO) and evaluate it on two medical data sets, one containing Chest X-ray Scans (CheXpert) and the other containing Retina Photographs (EyePACS). Both data sets are aimed at disease classification. We evaluate four different tasks: Linear Evaluation on In-Distribution Data, Linear-Evaluation on Out-of-Distribution Data, Semi-Supervised Evaluation on In-Distribution Data and Transfer Learning. We further benchmark the results of our method against various other methods with stochastic extensions as well as a baseline model. We show that Hierarchical-Masked-Attention is suited to improve Classification Performance on both data sets compared to competitor methods, while at the same time it is able to improve Model Calibration. Beyond that we show that Hierarchical-Masked-Attention can improve performance in Out-of-Distribution settings. For this, we chose two suitable Out-of-Distribution Data Sets (Chest-xray14 and APTOS), that provide a variety of Distribution Shifts. Future work could try to further explore our method's performance on medical data from different areas or make use of the model's stochastic component to incorporate methods of uncertainty qunatification or uncertainty quantification.

Contents

1. Introduction

Deep Learning based AI-methods have achieved impressive results across a variety of domains. Different model types excel in computer vision tasks such as image classification and image segmentation. Among the model types that deliver state-of-the-art results in classification tasks is the **Vision Transformer** [10]. With what seems like an ever-accelerating progress in predictive precision, questions about **secure** and **reliable** deployment of deep learning models in real life applications present themselves with a proportionally growing urgency. One area where not only predictive performance, but also a sense for the models shortcomings (i.e. how performance is affected by a shift of the distribution of the input data, or how much we are to trust the models output for a specific sample) is of utmost importance, is the area of healthcare and medicine, or, in the context of computer vision, **medical imaging analysis**. While researchers mostly focus (and succeed) on improving sota performance, model uncertainty often seems to be a topic that is neglected compared to its relevance. In order to achieve high predictive performance, as well as robust uncertainty measures, DL Models in general and the aforementioned ViT models in particular need large amounts of data to be trained on. DL models in computer vision usually utilize human-labelled image samples in a supervised setting as training data. However, labelling large amounts of images is financially and time-wise costly, even more so when working with medical images. A strategy to overcome this difficulty is to resort to self-supervised-learning learning rather than supervised learning. The goal of the thesis at hand is therefore to examine ViT models for medical imaging analysis in a self-supervised training setting, that achieve high predictive performance while addressing model calibration at the same time.

1.1 Medical Imaging Analysis

Deep Learning based AI applications can be applied to use cases in various medical fields, including, among others, radiology [29], dermatology[11], pathology [4]and ophthamology[9], and promise to match clinical experts in diagnosing diseases. However, there are some problems that are specific to medical imaging analysis which need to be addressed if we are to utilize the potential of DL algorithms in a beneficial way.

ID Generalization and OOD settings. Generalization is a primary challenge for medical imaging applications. Models in medical applications can be evaluated in either of two settings: *In-Distribution* and *Out-of-Distribution*. The latter refers to data that has been subject to a *Distribution Shift*, while the former has not undergone such a shift. In medical imaging, the most common Distribution Shifts are

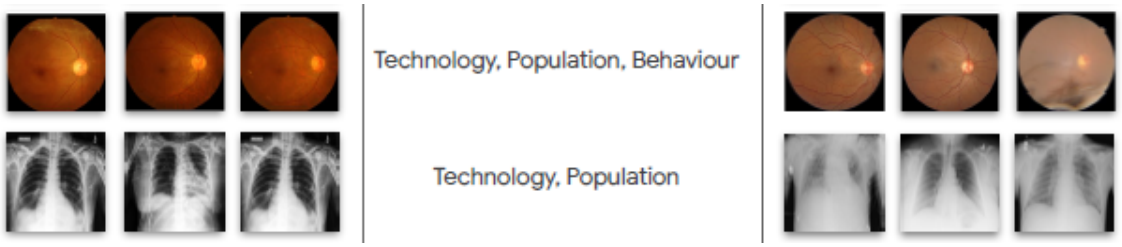


Figure 1 | Examples of Distribution Shifts. On the left visual samples of ID-Data are shown. In the middle the type of distribution shift is described. On the right samples of the OOD-Data are shown.

Figure Soure: [2]

technical shifts (e.g. a change of devices that are used to acquire data), population shifts (e.g. changes in demographic or clinical settings) and behaviour shift (e.g. changes in clinical practices) [12]. While models are able to reach performance levels close to human-experts on ID-data classification tasks, model accuracy has been observed to decline in OOD-settings[40, 41]. This finding poses a serious obstacle to safe and effective deployment of deep learning models in real life, as the maintenance of predictive performance under changed circumstances is elementary [21].

Generalization and Efficiency. The first idea that might come to mind when trying to tackle the problem of distribution shift, is to re-train the model on data from the shifted distribution.[8, 12]While this works well theoretically, it will be unfeasible in most real life cases, as data being labelled by clinical experts comes with high monetary expenses as well as a large amount of working hours. With an estimated annotation time of 122 seconds[32], and and estimated average wage of $205 per clinician, labelling a data set consisting of roughly 28.000 chest x-ray scans amounts to roughly $195.000 in spending and about 790 hours in clinician working hours [2]. Considering that the amount of data needed for training models in a supervised way is usually significant larger, efficient generalization remains a considerable problem.

1.2 Self-Supervised-Learning

Recent research has shown that the use of Self-Supervised-Learning can be beneficial in multiple ways. SSL methods start to approach the predictive performance of fully supervised models on common benchmark data sets, such as ImageNet[35], while they were shown to improve performance on OOD-Evaluation[16]and can thus be helpful in dealing with distribution shifts. At the same time, through SSLs' nature not being reliant on hard-labelled and human processed annotations of training data, the ability of effective generalization is increased significantly.

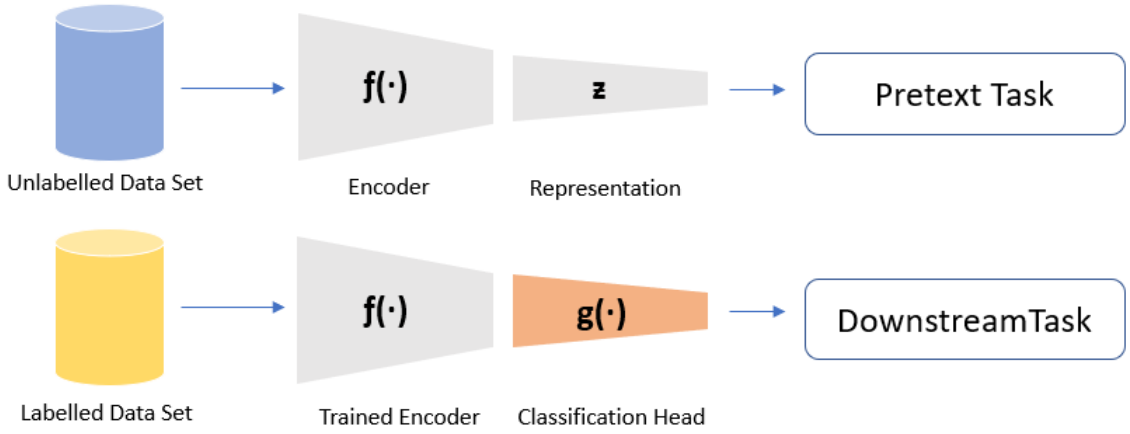


Figure 2 | Scheme of Self-Supervised-Learning. (top) The Encoder-Network $f(\cdot)$ is trained to solve pretext task on the unlabelled data set (bottom) The Linear Classifier $g(\cdot)$ takes z as inputs to solve downstream task

Concept of Self-Supervised-Learning. The general goal of a SSL framework is to train an encoder network $f(\cdot)$ on a pretext task $l_{pre}(x_{n1}, ..., x_{na})$ to learn a mapping for the input data x_n to a representation z_n, that in a second step, called the downstream task $l_{down}(x_n, y_n)$, can be utilized to learn a mapping from z_n to the corresponding label y_n for each x_n via a second function (usually a linear classification head) $g(\cdot)$. The final prediction for y_n presents itself as the composition $h = g \circ f$. Only the second step, the training of the classification head $g(\cdot)$, utilizes hard labels y_n, while the pretext task operates on a loss function that derives optimization targets from the unlabelled data set X by itself by design. There are a multitude of pretext tasks that can be used to learn the mapping to the representation space Z. Most recent and successful methods take a number of augmentations k of an input x_n, resulting in k views x_{na} of the same image x_n, with $a \in \{1, ..., k\}$.

Contrastive Loss. A well researched pretext task is contrastive loss, which found a popular adaption in SimCLR[7], defining a new standard for SSL when it was published. Intuitively and very briefly, SimCLR trains an encoder network through maximizing agreement between two augmented views of the same input image utilizing a constrastive loss function. However, there are a few disadvantages when utilizing contrastive loss as a pretext task: performance is heavily dependant on batch size, and therefore learning a good representation mapping is computationally expensive.

DINO. More recent research approaches try to adapt to this problem resulting in new SSL frameworks, one of which is DINO[5]. DINO managed to reduce parameters significantly, while also improving predictive performance. These are among the main reasons why this works has chosen DINO as the utilized SSL framework. A detailed description of its architecture is given in chapter 4.

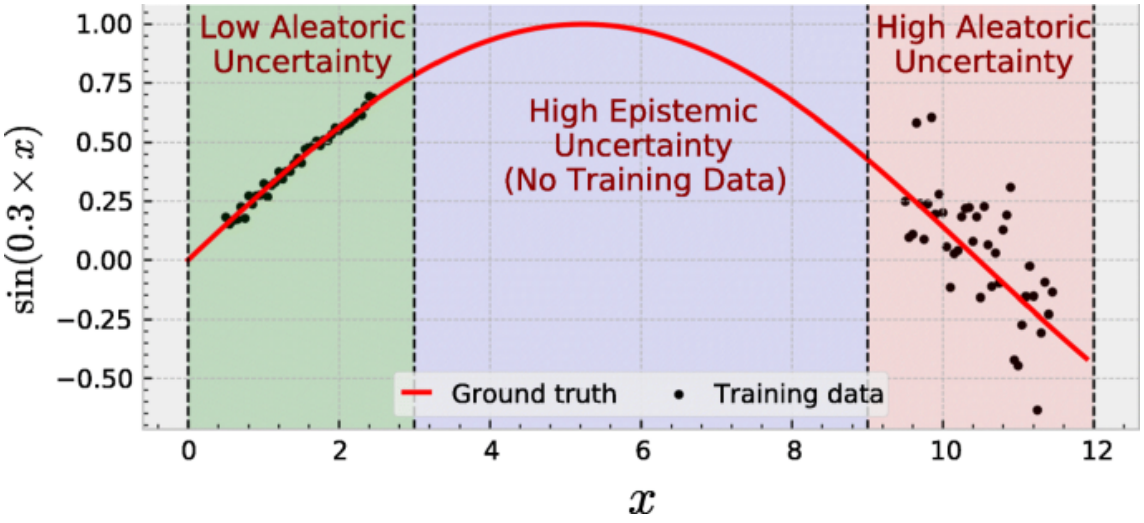


Figure 3 | 2D-Visualization of epistemic and aleatoric Uncertainty. In areas where no training data was seen, epistemic uncertainty is high. Aleatoric uncertainty is indiffeent to training data, but dependant on the data distribution itself.

Figure Soure: [37]

1.3 Uncertainty Estimation in DL Models

The predictive uncertainty of a DL model can be seen as composed of two parts, epistemic uncertainty and aleatoric uncertainty.

Epistemic Uncertainty. Epistemic uncertainty refers the component of uncertainty that is caused by the model itself. It can be interpreted as a probability distribution over the learned model parameters θ. [20]Therefore, epistemic uncertainty can be reduced with more training data that includes the desired information. There are various methods to capture epistemic uncertainty, of which many introduce a stochastic component into the models architecture which is then exploited to approximate the distribution of the parameters θ. We will examine different ways of introducing stochasticity into our chosen model architecture in later sections.

Aleatoric Uncertainty. Aleatoric uncertainy refers to the component of uncertainty that is not caused by the models parameters, but that rather is a property of the data distribution itself. It is therefore also known as Data Uncertainty and thus irreducible.[1]There are approaches tackling aleatoric uncertainty or entangling epistemic and aleatoric uncertainties, however this work focuses on stochastic approaches aimed at epistemic uncertainty.

2. Background and Related Work

As reliability and uncertainty estimation of DL models is an active field of research, there are numerous concepts and methods that are related to the work of this thesis, the two most important sub-fields being methods where a **Stochastic Component** is added to the transformer architecture on the one hand, and methods of estimating epistemic uncertainty through **Deep Ensembles** or **Bayesian Methods** on the other hand.

2.1 Stochastic Architectures in Transformer Models and SSL

Related work that has inspired the approach of this thesis are Pretrained Large Model Extensions [36], Hierarchical Stochastic Attention [30]and Sinkformers[31].

2.1.1 PLEX

PLEX is one of the most extensive studies of different reliability tasks that was published yet.

An extensive Study of Reliabilty. The aim of PLEX was to test the reliability of models, where a reliable model is seen as a model that not only achieves strong predictive performance, but one that also performs well across a variety of other tasks: tasks evaluated range across uncertainty tasks (e.g. selective prediction, open set recognition), robust generalization (e.g. accuracy, log-likelihood on ID and OOD evaulation) and adaption (e.g. active learning, few-shot uncertainty) and an extensive amount of different data sets from vision as well as language domains.

Stochastic Extension. In order to increase performance across these tasks, adaptions to the base architecture of ViT were made, including an approximation of the posterior distribution of last layer weights via a combination of a Gaussian Process Layer [25]and a Heteroscedastic last Layer [23]. The Gaussian Process layer is computed using SNGP, where the posterior presents itself as:

$$g(x) \sim N(logit(x), var(x)) \tag{2.1}$$

where $logit(x)$ is a random feature approximation ϕ applied to the predictive function and $var(x)$ is a Laplace-Approximation of the predictive variance.

Pretrained Weights and Finetuning. PLEX uses different large pretrained models and fine tunes and evaluates them on downstream tasks.

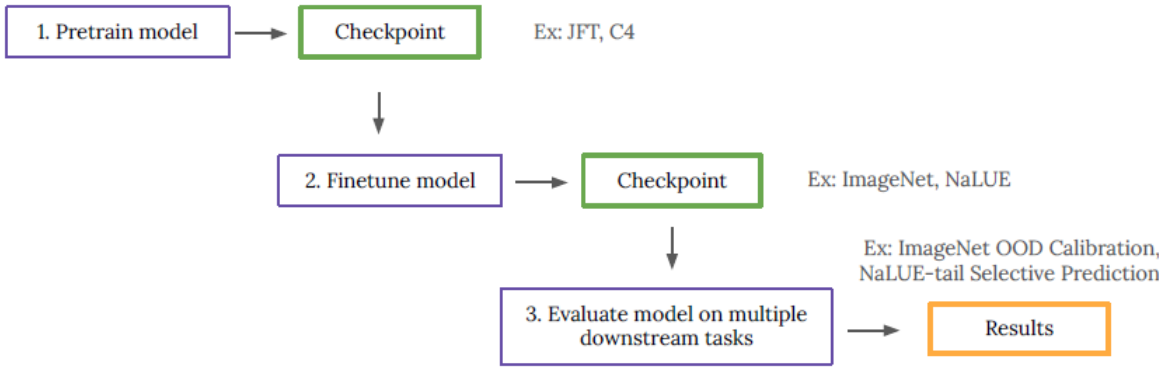


Figure 4 | Plex Training Pipeline.

Figure Soure: [36]

2.1.2 Hierarchical Stochastic Attention through Gumbel Softmax

A straightforward and effective strategy that retains predictive performance and enables the network of Uncertainty or Reliablity Estimation is to directly alter the attention mechanism.

Replacing Softmax with Gumbel Softmax. The standard design of ViT uses a Multi-Head-Attention mechanism that computes an attention score a_i for each *i-th* Attention Head. We can easily induce stochasticity through replacing the Softmax-Function of the Attention computation with the Gumbel-Softmax-Function [15], resulting in a stochastic Attention score:

$$\hat{a}_i = \frac{\mathcal{G}(q_i k_i^T)}{\tau} \tag{2.2}$$

Hierarchical Extension. This technique can further be extended into a hierarchical Attention mechanism, where an intermediate step utilizing a learnable centroid vector C is introduced into the computations of Key, Query and Value. Through this, the model is supposed to learn to pay attention to the centroids. In this work, we consider experiments with both, the simple change in Attention as well as the hierarchical method involving centroids.

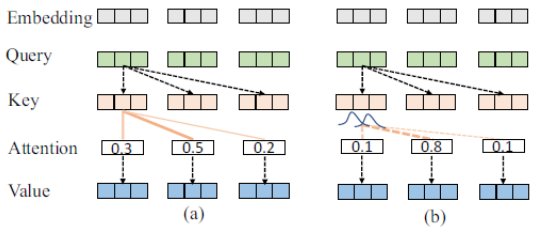


Figure 5 Visual Comparison of standard and Gumbel-Softmax Attention

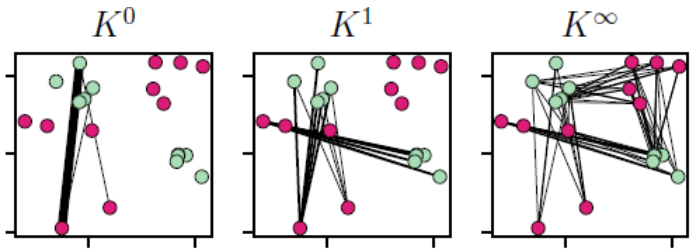


Figure 6 | Illustration of the diferent Normalizations of Attention Matrices. Applying Normalization through Sinkhorn's Algorithm. For K^∞ (Sinkhorn), all Data Points are involved in an interaction.

Figure Soure: [31]

2.1.3 Double Stochastic Attention with Sink Formers

The learnable Attention Matrix of standard Transformer models such as ViT involve pairwise interaction between Data Points. This matrix is normalized with the Softmax-Function, resulting in a row-wise stochasticity of the matrix.

Double Stochasticity. Through application of Sinkhorn's Algorithm [33], stochasticity can be introduced not only row-wise but also column-wise, making the Attention Matrix effectively double-stochastic. Compared to the standard Softmax-Operation, normalization through Sinkhorn's Algorithm considers more interactions between data points, as all points are matched to one another with different degrees of intensity. Double stochasticity has been shown to improve predictive performance, however the authors are not conducting a study of reliability or uncertainty.

2.2 Uncertainity Estimation

Uncertainty Estimation methods can be subdivided into two fields: Ensemble-Methods and Bayesian Neuronal Networks. For the experiments of this work concepts stemming from both are considered.

2.2.1 Deep Ensembles

In Deep Ensembles [24], multiple neuronal Networks are trained and the final prediction is retrieved from their individual predictions. Deep Ensembles usually deliver SOTA predictive performance. Their big disadvantage lies in their computational cost, as each ensemble member requires it's own full training run. Given a set of ensemble members' posteriors $\{P(y|x,\theta^{(m)})\}_{m=1}^{M}$, where m is the number of members, the amount of disagreement between the Ensemble members can be utilized to produce an estimation of model uncertainty, e.g. via entropy calculation:

$$\underbrace{MI[y,\theta|x^*]}_{Knowledge\,Uncertainty} = \underbrace{H[\mathbb{E}_{p(\theta|D)}[P(y|x^*,\theta)]]}_{Total\,Uncertainty} - \underbrace{\mathbb{E}_{p(\theta|D)}[H[P(y|x^*,\theta)]]}_{Expected\,Data\,Uncertainty} \tag{2.3}$$

where x^* is a sample of a data set D.[1]

2.2.2 Bayesian Neuronal Networks

The Bayesian Neuronal Networks approach is, contrary to the frequentist approach that aims for parameter point estimates, to define a full probability distribution over the network parameters. This posterior distribution, that semantically represents the belief or uncertainty about the value of each parameter, presents itself through marginalizing over all parameter settings as:

$$\int p(D|w)p(w)\,dw \tag{2.4}$$

[39]

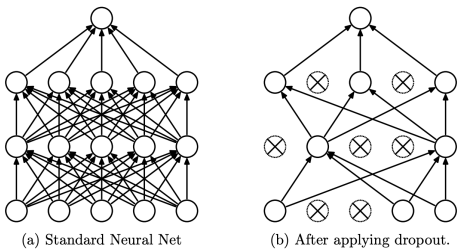
where D is a Data Set and w are the network weights, which results via Bayesian Model Averaging in a full predictive distribution:

$$P(y|D,x) = \int p(y|w,x)p(w|D)\,dw \tag{2.5}$$

[39]

that defines the probability for class label y given new input x. Bayesian Neuronal Networks can therefore be used to obtain a more realistic estimation of network uncertainties and network calibration than frequentist approaches. However, the integral of $P(y|D,x)$ is often intractable for neuronal networkas, due to its complexity, and thus needs to be approximated. [3]

Monte Carlo-Dropout. One common and conceptually easy approach to approximating the posterior is Monte Carlo-Dropout.[14]Dropout provides stochasticity or variation in a neural network by randomly shutting down weights during training. It can be reinterpreted as approximate Bayesian inference and applied during testing, which leads to multiple different parameter settings. This can be utilized to do multiple forward passes of the same sample x to give an estimation of uncertainty.[3]



(a) Standard Neural Net (b) After applying dropout.

Figure 7 | Illustration of Dropout. In MC-Dropout, dropout is applied during training as well as testing.

Figure Soure: [34]

3. Method: Hierarchical-Masked-Attention

This work examines a Vision-Transformer architecture [10]in a Self-Supervised-Setting, at the core of which lies a Multi-Head-Attention mechanism[38]. We propose a stochastical extension of the Attention mechanism through masking a number of heads during each forward pass, combined with a scheduler that works to thin out the proportion of masking that is applied over the course of the training.

3.1 Multi-Head-Self-Attention

Self-Attention. The Attention Score of an input is calculated via the Softmax-Score of the dot product of a Key and Query Matrix, which are divided by a scaling factor, multiplied with the Value Matrix, which represents itself as:

$$Attention(Q, K, V) = SOFTMAX(\frac{QK^T}{\sqrt{d_k}})V \tag{3.1}$$

where Q, K and V are learnable linear transformations. As the structure of the mechanism refers K, Q and V from the same input, it is called *Self*-Attention, which could, loosely speaking, semantically be interpreted as data learning to pay attention to itself through the Transformer Encoder.

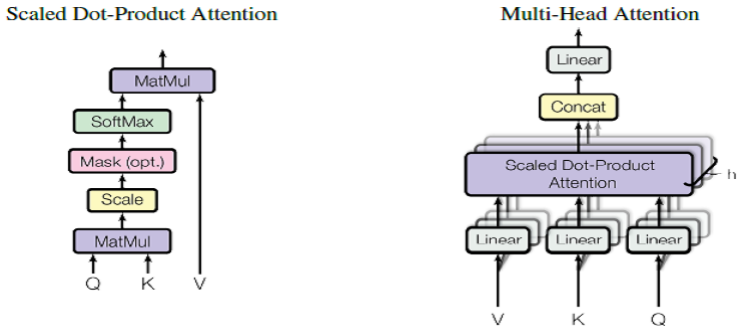


Figure 8 | Multi-Head-Attention. (Left) Visualization of a single Attention Head computation (Right) Visualization of multiple Attention Heads running parallel.
Figure Soure: [38]

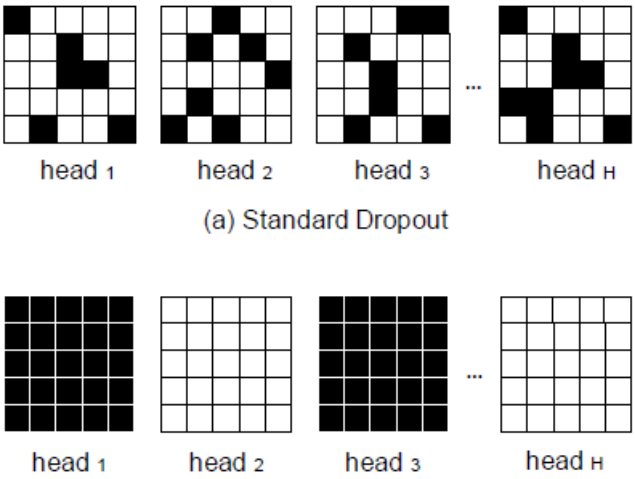


Figure 9 | Masked Attention. Visual comparison of standard Dropout (Top) and masking Attention Heads (Bottom.

Figure Soure: [42]

Multi-Head-Self-Attention.

In practice, most Vision Transformers use H heads (typically between 6 and 16), where each head creates and applies its own Attention Matrix resulting from distinct Q, K and V matrices [38]. Thus Multi-Head-Attention is calculated as:

$$Attention(Q_h, K_h, V_h) = SOFTMAX(\frac{Q_h K_h^T}{\sqrt{d_k}})V_h \tag{3.2}$$

where h $\in \{1, ..., H\}$. Multi-Head-Attention is what lies at the heart of this work. We evaluate standard Multi-Head-Attention as well as different extensions of it that introduce stochasticity.

3.2 Hierarchical-Masked-Attention

The extension this work proposes that is original to it, and that will be benchmarked against other methods, can be referred to as **Hierarchical-Masked-Attention**.

Masked-Attention. In its core, it masks a number m of attention heads during every forward pass. A similar concept has been shown to work as an effective regularizer and be beneficient to model performance.[42]

Hierarchical-Masked-Attention. Additionally, a scheme is added that thins out the numbers of masks applied throughout the course of training. For this, a Hyperparamter $h_p \in [0, 1]$ is added that acts as a threshold. Each time a forward pass is computed, a random number between $[0, 1]$ is sampled. M masks are applied to

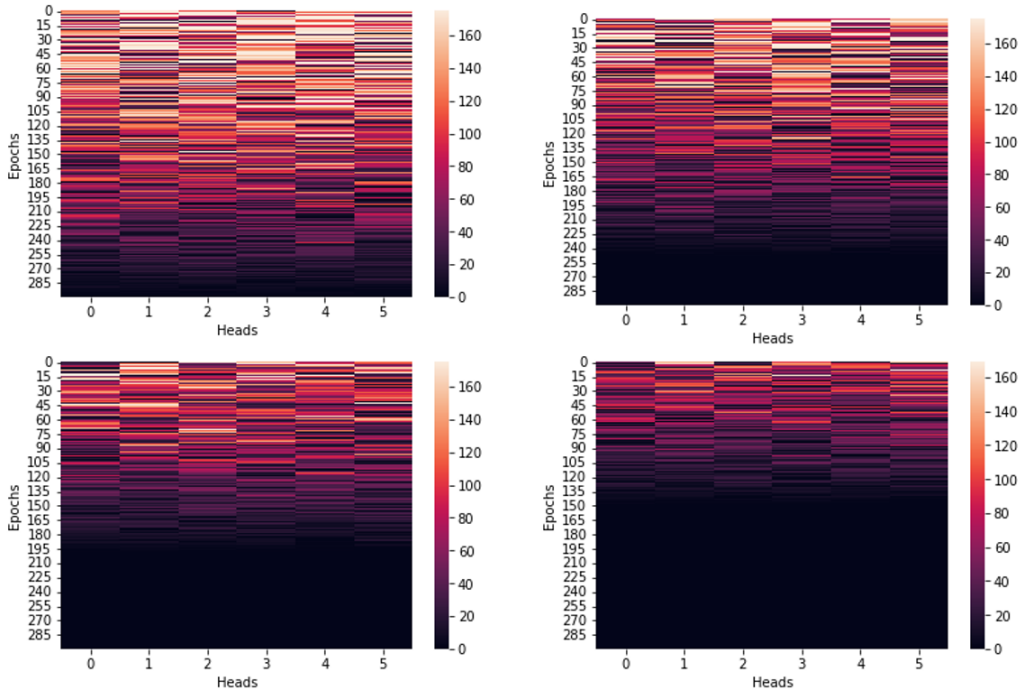


Figure 10 | Hierarchical-Masked-Attention. As the threshold lowers over progressing through training epochs, each head is masked less. The effect of different values for h_p ranging from 1.5 (Top Left) to 0.75 (Bottom Right) over 300 epochs are displayed. Note that in order to make the effect more visible, the oversampling threshold T is not applied in this schematic graphs.

randomly selected heads, if the sampled random number is below the threshold h_p. After each epoch, the threshold is lowered by 0.005. An additional Hyperparameter T controls how often a single head can be masked, in order to ensure that the sampling spreads out equally among all heads and a single head will not be oversampled. This scheme results in an inverted-pyramid shape of the number of masks applied over training time, with the amount of masking for each individual head thinning out. A description of the algorithm in pseudo code is given below:

Algorithm 1 Hierarchical-Masked-Attention

Require:

$h_p >= 0$ ▷ Thinning Parameter
$T > 0$ ▷ Oversampling Parameter
$m > 0$ ▷ Number of Masks to be applied
$v_1, ..., v_H$ ▷ Value Outputs of each Head of current Epoch

1: **for** `each epoch` **do**
2: $combs \leftarrow \{1, .., H\}_1 \times ... \times \{1, .., H\}_m$ ▷ List of all Sample Combinations

sample_counts $\leftarrow [0, .., 0]$ of len(*combs*) ▷ List of 0s

for `each forward pass` **do**
  $p \leftarrow U[0, 1]$

  if $p < h_p$ **then**
    `sample` m `elements from` *combs* ▷ postions to mask
    sample_counts `at indices m` $= +1$
    `mask` v_h `at sampled positions` m ▷ mask

    if *sample_counts* at indices $m >= T$ **then**
      `remove combination from` *combs*
    end if
  end if
end for
$h_p = h_p - 0.005$ ▷ lower thinning parameter
end for

The general thought behind the hiearchical application of masks is to encourage the model to learn robust representations through the induction of stochasticity in the early stages of the training procedure, and after it has supposedly done so, enable it to further improve them without the stochasticity counteracting the learning progress of the early stages.

4. Self-Supervised-Learning Framework and Implementation

This work aims to apply the method described under chapter 3 as well as benchmark it against similar concepts in a self supervised setting. The Self-Supervised-Learning frame we have chosen is DINO [5].

4.1 Network Architecture

Dino utilizes Knowledge Distillation[18]between a student model s and a teacher t model to learn representation mappings.

4.1.1 DINO

Architecture. Student and Teacher both share same the network architecture g, which is composed of a backbone model f and a projection head h, such that $g = h \circ f$. The backbone f can be a Transformer Model (ViT in our case) or a ResNet[17], the projection head h consists of a 3-Layer Multi-Layer-Perceptron with hidden dimension 2048 followed by l_2 Normalization and a weight normalized fully connected layer[5].

Forward Passes and Knowledge Distillation. For each input image, a set V of different augmented views is generated. It contains two global views, x_g^1 and x_g^2 and several local views of smaller resolution. All crops are passed through the student while only the global views are passed through the teacher, therefore encouraging "local-to-global" correspondences [5].

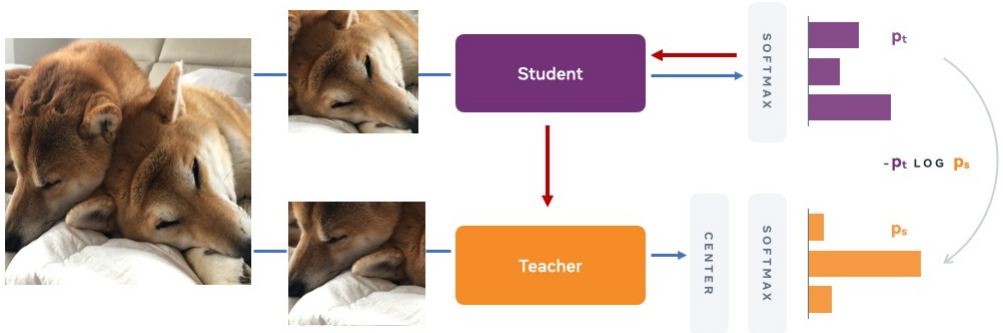


Figure 11 | Schematic Dino Architecture. Blue arrows mark forward pass, red arrows mark backpropagation.

Figure Soure: [5]

The output of both networks are normalized with a Softmax-Function, after which the loss between the resulting distributions $P_t(x)$ and $P_s(x')$ is minimized as follows:

$$\min_{\theta_s} \sum_{x' \in \{x_1^g, x_2^g\}} \sum_{\substack{x' \in V, \\ x' \neq x}} H(P_t(x), P_s(x'))$$

where H is the cross entropy:

$$H(a, b) = -a \; log \; b \tag{4.1}$$

Note that the cross entropy is minimized w.r.t only the student network's parameters θ_s. The teacher network's parameters θ_t are updated in dependance to the student's parameters. Details are given under 4.2. Also note that no data set given target labels are involved. The resulting output distributions which are used to calculate loss values are virtual K-dimensional distributions with no inherent meaning.

4.1.2 Vision Transformer

Our backbone model f is a Vision Transformer [10].

ViT Overview. Vision Transformers divide an input picture into multiple patches, to each of which a linear projection is applied. These embedding projections are then passed into the encoder structure, alongside an additional learnable positional embedding. The latter's function can be interpreted as equivalent to BERT's [26]CLS-token. The encoder structure is composed of multiple layers of a Multi-Head Attention mechanism followed by a MLP, with Normalization applied before both. The concatenated output of the final Encoder Layer or of the n last Encoder layers are again processed through a MLP-Head, which produces the final output embeddings that can further be utilized for classification or other tasks. Our method described under 3.2 is applied in the Multi-Head Attention mechanism within the Encoder architecture.

Specifications. Specifically, we use ViT-s for the experiments presented in this work. Important architectural parameters were set as follows:

- Patch Size of 16x16 pixels.
- Embedding Dimensions of 384.
- Depth of 12.

Patch Size determines the size of the input patches. Smaller Patch Sizes can achieve better performance, but are also computationally more expensive. Embedding Dimension determines the size of projection outputs. Depth determines how many subsequent Multi-Head Attention layers are utilized. ViT-s amounts to a total of 21M parameters, which makes it comparatively small.

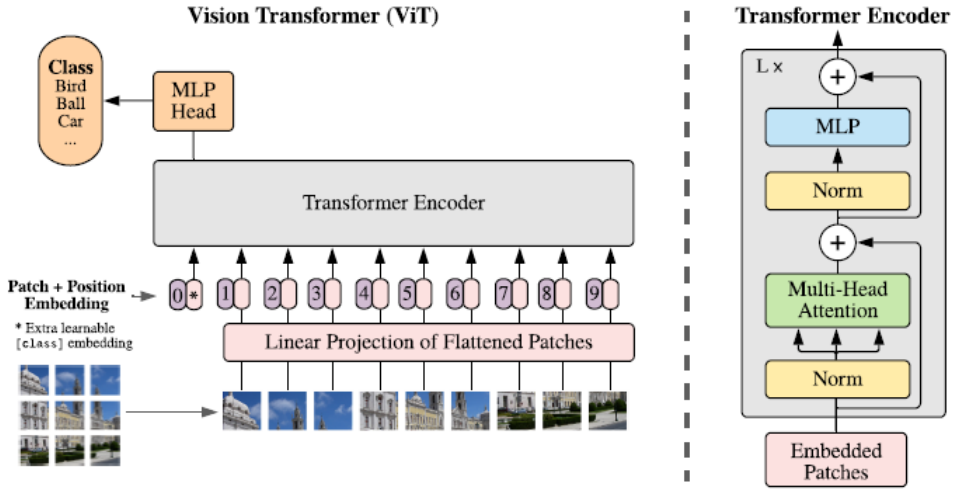


Figure 12 | ViT Architecture. (Left) Schmematic Forward Pass (Right) Transformer Encoder Architecture

Figure Soure: [10]

4.2 Optimization

Optimization and Training Parameters. The network was optimized using ADAMW[28]with a batch size of 64 distributed over 8 GPUS which results in an effective batch size of 512. The learning rate was ramped up linearly to its target value of 0.0005 during an warm up of 10 epochs, as proposed by the authors of DINO. After the warmup the learning rate is decayed following a cosine schedule [27]. The weight decay also follows a cosine schedule starting from 0.04 to 0.4. Other parameters, if not stated otherwise, are kept as DINO default values.

Teacher Network Updates. As no teacher network g_t is given a priori, it is constructed during training from the student network. The weights θ_s of the student network g_s are learned through backpropagation, while the weights of the teacher network θ_t are inferred from past iterations of the student through an exponential moving average with the following update rule:

$$\theta_t \leftarrow \lambda\theta_t + (1-\lambda)\theta_s \tag{4.2}$$

where λ follows a cosine schedule from 0.996 to 1 over the course of the training. All evaluations are done on weights of the teacher model.

Augmentations. As proposed by DINO authors Color Jittering, Gaussian Blur and Solarization were used as augmentations. For details on cropping views refer to the paper and/or the official GitHub Repository.

5. Experiments

This work evaluates and benchmarks the proposed Self-Supervised Framework on two medical Data Sets and a variety of downstream tasks, which are described in the following.

5.1 Datasets

We consider CheXpert[22], a Data Set of Chest X-ray scans, and EyePACS[13]a Data Set of retina images. For both Data Sets, a respective OOD Data Set is considered to evaluate model performance under distribution shift. For CheXpert, Chest-xray14[6]is used as OOD-Data Set, for EyePACS, the APTOS Data Set[19]is considered. Both Data Sets contain the same types of diseases and class labels.

5.1.1 CheXpert

CheXpert is a large public Data Set for chest radiograph interpretation, consisting of 223.648 chest radiographs of 65.240 patients. It was retrospectively collected by the chest radiographic examinations from Stanford Hospital, performed between October 2002 and July 2017[22]. It contains labels for the presence of 14 observations, each amounting to a different lung disease, labeled as 0 (=negative), 1 (=positive) or -1 (=uncertain). CheXpert contains frontal as well as lateral views. Following the CheXpert evaluation protocol[22], only 5 out of 14 disease labels will be used to evaluate model performance. See 5.2.1 for details. The training set consists of 223.414 samples, the testing set of 234 samples. CheXpert is available in two resolutions, we use the variant with lower resolution. Images are sized around 380x320 pixels.

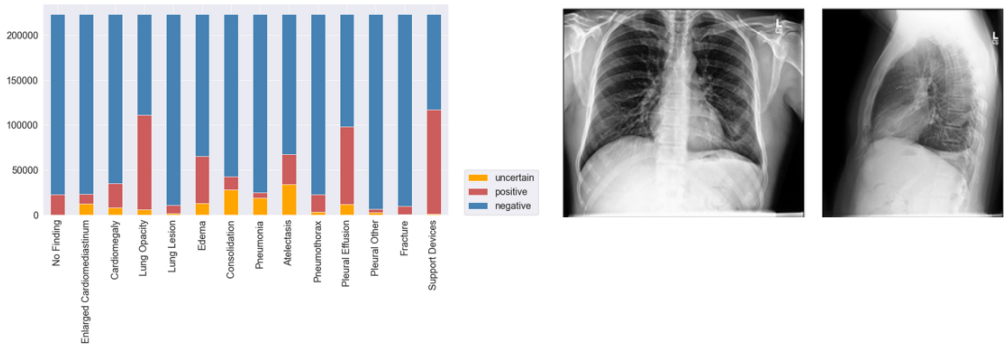


Figure 13 | CheXpert Data. (Left) CheXpert Label Distribution over Classes. (Right) Frontal and lateral scan samples of CheXpert Data.

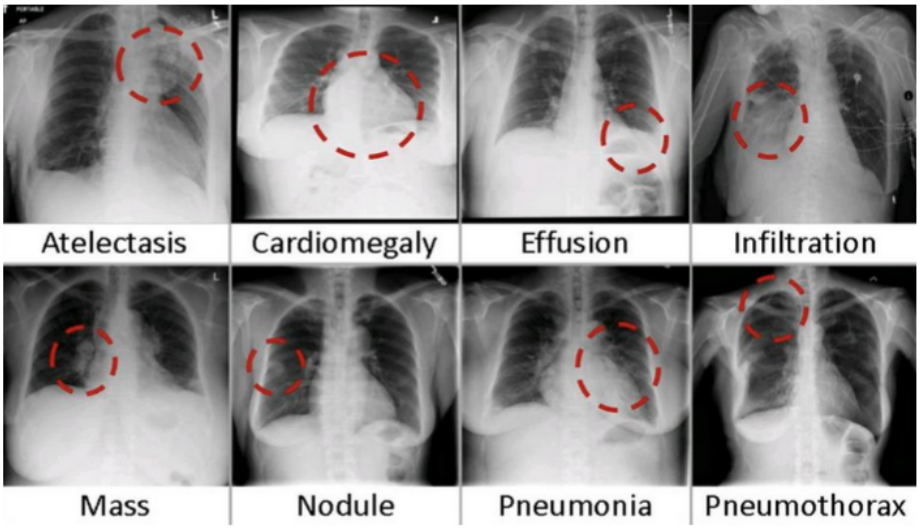


Figure 14 | Chest-xray14 Data. Chest-xray14 samples of different diseases.
Figure Soure: [6]

5.1.2 Chest-xray14

Chest-xray14 is a Data Set of chest x-ray scans similiar to CheXpert, consisting of 112.120 frontal-view X-ray images of 30.805 patients, collected from the year of 1992 to 2015. It contains labels for the same 5 disease classes that CheXpert is evaluated on, and was collected in different clinics and with different scanning technology, which makes it suitable to evaluate our model in an OOD-setting under population and technology shift. Labels are given as 0 (=negative) and 1 (=positive).

5.1.3 EyePACS

EyePACS is a large set of high-resolution digital color Fundus Photographs (given in different sizes, with the largest around 3900x2500 pixels) of the Retina taken under a variety of imaging conditions. It contains labels for different severity levels of Diabetic Retinopathy, which is an eye disease associated with long-standing diabetes. Labels are given for each image on a scale of 0 to 4, according to the following scale:

- 0 = No DR
- 1 = Mild
- 2 = Moderate
- 3 = Severe
- 4 = Proliferative DR

The Data Set consists of a total of 88.702 colour Fundus Images, including 35.126 samples for training and 53.576 for testing.

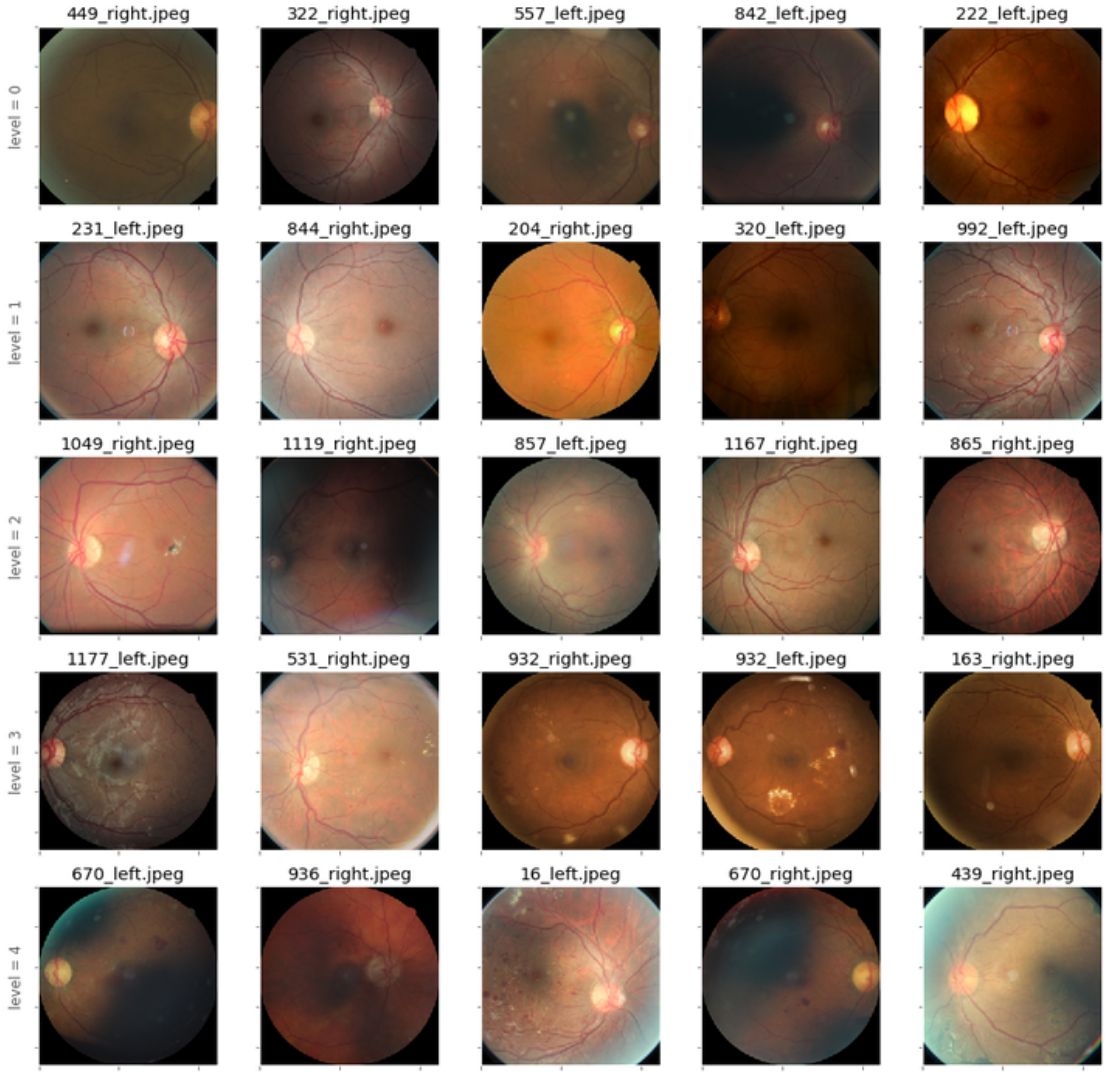


Figure 15 | EyePACS Data. Samples of EyePACS data. Each rows shows a different level of severity, increasing from Top(=0) to Bottom(=4).

5.1.4 APTOS

APTOS is a Data Set that also contains Retina Photographs, with the same labels and scale for Diabetic Retinopathy as EyePACS. It consists of 3663 images, and, as it was collected by the Aravind Eye Hospital in India, can therefore be used to evaluate out model in an OOD-setting under population shift, technology and behaviour shift.

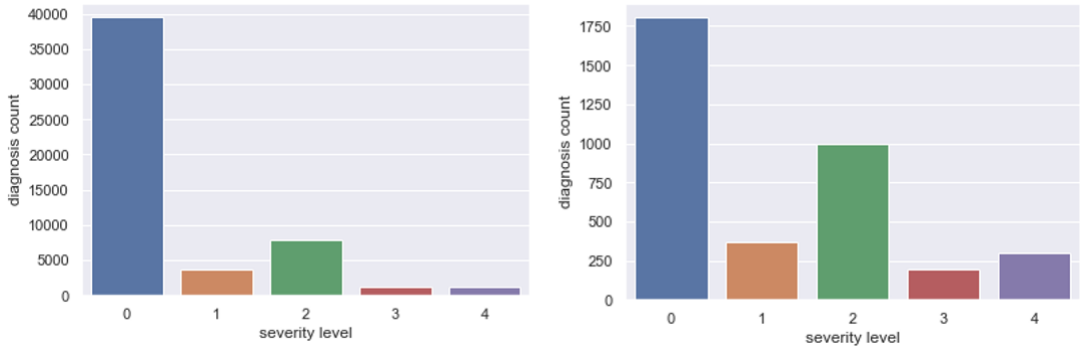


Figure 16 | Label Distribution of both Retina Data Sets. (Left) EyePACS labels (Right) APTOS labels

5.2 Downstream Tasks

We evaluate model performance on 4 different tasks:

1. Linear Evaluation on In-Distribution Data
2. Linear Evaluation on Out-of-Distribution Data
3. Semi-Supervised Evaluation
4. Transfer Learning

5.2.1 Linear Evaluation on In-Distribution Data

For Linear Evaluation, we load and freeze the weights of the ViT that was trained as backbone model f during Self-Supervised-Training. The outputs of the Transformer are passed into a Linear Classifier, which produces the final classification outputs through a Sigmoid-Layer. The Linear Classifier is fine tuned supervised on training data labels for 10 epochs with the Loss Function BCEWithLogitsLoss which is suitable for both data sets, if the number of targets are adjusted respectivley. Note that weighting the loss function with class ratios was **not** beneficial during multiple test runs.

CheXpert Task. CheXpert is evaluated on Multi-Label-Classification, as each data sample can contain multiple non-exclusive disease diagnoses. Following the 'U-Ones' strategy of the CheXpert evaluation protocol, we set all uncertainty labels (=-1) to 1 for the finetuning of the Linear Classifier. Furthermore, of CheXperts 14 target labels only 5 are considered to evaluate performance: Cardiomegaly, Edema Consolidation, Atelectasis and Pleural Effusion.[22]

EyePACS Task. For EyePACS evaluation, we use the same approach as PLEX [36]does for evaluating the country shift task: the Data Set's 5 labels are used to assign a new binary label as follows:

- 0 = mild (original labels 0 and 1)
- 1 = severe (original labels 2,3 and 4)

We then evaluate Binary-Classification based on the newly assigned labels.

Metrics. For CheXpert, we report AUROC for each individual target label as well as the average of all 5 target labels. Also reported are the Expected Calibration Error of each individual label as well as an average of all 5 target labels, and the Negative Log-Likelihood. For EyePACS, we report Binary Accuracy, as well as ECE and NLL.

5.2.2 Out-of-Distribution Data

For OOD-Evaluation, the test sets are replaced with their respected OOD Data Sets, as described in 5.1.2 and 5.1.4. Note that we do Zero-Shot-OOD-Evaluation, as proposed in [2], which means that our model is neither trained nor fine tuned on any OOD-data. For this task, only AUROC and Binary Accuracy are reported.

5.2.3 Semi-Supervised Evaluation

For Semi-Supervised-Evaluation, we follow the exact same approach as for Linear Evaluation in 5.2.1, expect we only use 10% and 1% of the training data to fine tune the linear classifier. For this task, again only AUROC and Binary Accuracy are reported.

5.2.4 Transfer Learning

To evaluate the models capability under Transfer Learning, we followed the same training procedure as for the Baseline Model (i.e. no stochasticity added) but initialized the DINO-network with weights that were pretrained on ImageNet as released by the authors of DINO. AUROC/Binary Accuracy, ECE and NLL are reported.

5.3 Competitors

We benchmark our method proposed in 3 against a Baseline Model as well as other stochastic extensions of ViT.

5.3.1 Baseline

For a Baseline comparison, we evaluate both Data Sets on unchanged training runs of DINO under the conditions specified in 4.2.

5.3.2 Ensemble

We evaluate the performance of an ensemble of 3 individually trained ViTs. The final prediction is obtained through averaging the outputs of the individual models before passing to the Linear Classifier.

5.3.3 Monte-Carlo-Dropout

We evaluate a Monte-Carlo extension of the DINO network. As described in 2.2.2, Dropout is kept during evaluation.

For CheXpert, we consider:

- Dropout-Rate 0.1
- Dropout-Rate 0.15

For EyePACS, we consider:

- Dropout-Rate 0.15

5.3.4 Gumbel Softmax

We evaluate a Gumbel-Softmax extension of the attention mechanism as described in 2.1.2.

For CheXpert, we consider:

- Gumbel Softmax Stage 1, $\tau = 1$
- Gumbel Softmax Stage 1, $\tau = 3$
- Gumbel Softmax Stage 1, $\tau = 5$
- Gumbel Softmax Stage 1, $\tau = 20$
- Gumbel Softmax Stage 2, $\tau_1 = 5$, $\tau_2 = 20$
- Gumbel Softmax Stage 2, $\tau_1 = 30$, $\tau_2 = 20$
- Gumbel Softmax Stage 2, $\tau_1 = 1$, $\tau_2 = 20$

For ExePACS we consider:

- Gumbel Softmax Stage 1, $\tau = 1$

Where Stage 1 refers to the replacement of the Softmax-Function in the Attention mechanism, and Stage 2 refers to the hierarchical extentions including a centroid. Note that due to comparatively weak results Stage 2 was omitted for EyePACS. For EyePACS only a setting of τ=1 was considered due to time and computational limitations. The τ-s were chosen similar to the experiments of the original paper[30].

5.3.5 Hierachical Masked Attention

We evaluate our proposed method for multiple settings of the number of masks m and the thinning parameter h_p, as well as for an increasment of heads:

For CheXpert, we consider:

- $m = 1$, $h_p = 0.5$
- $m = 2$, $h_p = 0.5$
- $m = 3$, $h_p = 0.5$
- $m = 4$, $h_p = 0.5$
- $m = 1$, $h_p = 0.1$
- $m = 2$, $h_p = 0.2$
- $m = 1$, $h_p = 0.1$
- $m = 2$, $h_p = 0.2$
- $n_{\text{heads}} = 16$, $m = 4$, $h_p = 0.5$
- $n_{\text{heads}} = 16$, $m = 6$, $h_p = 0.5$
- $n_{\text{heads}} = 16$, $m = 4$, $h_p = 0.1$
- $n_{\text{heads}} = 16$, $m = 6$, $h_p = 0.1$

For EyePACS we consider:

- $m = 1$, $h_p = 1.5$, $epochs = 300$
- $m = 1$, $h_p = 0.75$ $epochs = 300$
- $m = 1$, $h_p = 0.35$ $epochs = 300$
- $m = 2$, $h_p = 1.5$, $epochs = 300$
- $m = 1$, $h_p = 4$, $epochs = 800$
- $m = 1$, $h_p = 12$, $epochs = 2400$

Training Time and Effect of h_p. Note that, in the case of 100 training epochs $h_p = 0.5$ amounts to the masking of Attention Heads being applied over the whole course of training, reaching 0 just as training finishes. The same is true for the respective cases of h_p being equal to 1.5, 4 and 12 for 300, 800 and 2400 training epochs. Lower h_ps therefore means that masking is only applied during a fraction of training time.

6. Results

Below the results of the above described experiments for all tasks mentioned in 5.2 are presented.

6.1 CheXpert

In each table the Top 3 performing models are highlighted. Blue marks AUROC, Red marks ECE and Green marks NLL.

6.1.1 In-Distribution Evaluation

Experiment	Average	Cardiomegaly	Edema	Consolidation	Atelectasis	Pleural Effusion
Baseline, Seed 1	0.8654	0.8150	0.9283	0.9059	0.7815	0.8965
Baseline, Seed 2	0.8599	0.8456	0.9137	0.8992	0.7878	0.8534
Baseline, Seed 3	0.8534	0.8116	0.9141	0.9220	0.7785	0.8406
Baseline, 300 Epochs	0.8562	0.8150	0.9150	0.9042	0.7512	0.8957
Ensemble (100,100,100)	0.8630	0.8442	0.9132	0.9220	0.7948	0.8406
Ensemble (300,100,100)	0.8713	0.8398	0.9386	0.9093	0.7785	0.8903
Vit Base	0.8618	0.7959	0.9399	0.9119	0.8148	0.8464
MC-Dropout (0.1)	0.8450	0.7789	0.9043	0.9186	0.7971	0.8261
MC-Dropout (0.15)	0.8339	0.7776	0.8950	0.9034	0.7822	0.8116
MC-Dropout (0.2)	0.8507	0.8058	0.9172	0.9186	0.7692	0.8427
G-Softmax I, $\tau = 1$	0.8196	0.7680	0.8936	0.8678	0.7898	0.7789
G-Softmax I, $\tau = 3$	0.8255	0.7265	0.9114	0.8949	0.7576	0.8369
G-Softmax I, $\tau = 5$	0.8331	0.7782	0.8789	0.8500	0.7995	0.8588
G-Softmax I, $\tau = 20$	0.6770	0.6364	0.6337	0.7754	0.6322	0.7072
G-Softmax II, $\tau_1 = 1$, $\tau_2 = 20$	0.7488	0.6857	0.7815	0.8364	0.7090	0.7313
G-Softmax II, $\tau_1 = 5$, $\tau_2 = 20$	0.7508	0.6544	0.7988	0.8008	0.7233	0.7764
G-Softmax II, $\tau_1 = 30$, $\tau_2 = 20$	0.7587	0.6704	0.8371	0.8042	0.7233	0.7582
HM-Attention, $m = 1$, $h_p = 0.5$	0.8591	0.8235	0.9217	0.9169	0.8094	0.8240
HM-Attention, $m = 2$, $h_p = 0.5$	0.8698	0.8405	0.9243	0.9229	0.8151	0.8460
HM-Attention, $m = 3$, $h_p = 0.5$	0.8500	0.8364	0.9150	0.8975	0.7885	0.8128
HM-Attention, $m = 4$, $h_p = 0.5$	0.8508	0.8286	0.9088	0.9110	0.7788	0.8300
HM-Attention, $m = 1$, $h_p = 0.1$	0.8609	0.8252	0.9230	0.9059	0.7955	0.8551
HM-Attention, $m = 1$, $h_p = 0.2$	0.8525	0.8180	0.9194	0.8822	0.8021	0.8406
HM-Attention, $m = 2$, $h_p = 0.1$	0.8450	0.7827	0.8945	0.9314	0.7948	0.8215
HM-Attention, $m = 2$, $h_p = 0.2$	0.6407	0.5656	0.6435	0.7568	0.5361	0.7014
HM-Attention, $n_{\text{heads}} = 16$,$m = 4$, $h_p = 0.5$	0.8574	0.8061	0.9217	0.8958	0.8035	0.8600
HM-Attention, $n_{\text{heads}} = 16$,$m = 6$, $h_p = 0.5$	0.8433	0.8190	0.8936	0.9076	0.8138	0.7826
HM-Attention, $n_{\text{heads}} = 16$,$m = 4$, $h_p = 0.1$	0.7446	0.7337	0.8696	0.8194	0.6861	0.6145
HM-Attention, $n_{\text{heads}} = 16$,$m = 6$, $h_p = 0.1$	0.8325	0.8313	0.8847	0.8839	0.7483	0.8145

Table 6.1 AUROC Scores for CheXpert ID-Evaluation

Experiment	Average	Cardiomegaly	Edema	Consolidation	Atelectasis	Pleural Effusion	NLL
Baseline, Seed 1	0.1133	0.1261	0.0961	0.0887	0.1022	0.1532	0.7270
Baseline, Seed 2	0.0987	0.1607	0.0722	0.0853	0.0631	0.1121	0.7272
Baseline, Seed 3	0.0993	0.1221	0.0660	0.0779	0.0962	0.1340	0.7278
Baseline, 300 Epochs	0.0960	0.1220	0.0566	0.0685	0.0736	0.1593	0.7268
Ensemble (100,100,100)	0.1030	0.1474	0.0890	0.1017	0.0664	0.1108	0.7279
Ensemble (300,100,100)	0.1041	0.1257	0.0877	0.0794	0.0895	0.1381	0.7260
Vit Base	0.1170	0.1337	0.0938	0.1235	0.0919	0.1421	0.7261
MC-Dropout (0.1)	0.0994	0.1179	0.0741	0.0823	0.0895	0.1335	0.7291
MC-Dropout (0.15)	0.0989	0.1164	0.0766	0.0929	0.0779	0.1307	0.7324
MC-Dropout (0.2)	0.0934	0.1149	0.0853	0.0873	0.0645	0.1151	0.7284
G-Softmax I, $\tau = 1$	0.1033	0.1206	0.0616	0.1040	0.0901	0.1401	0.7344
G-Softmax I, $\tau = 3$	0.0987	0.1265	0.0646	0.0851	0.0748	0.1425	0.7346
G-Softmax I, $\tau = 5$	0.1066	0.1213	0.1213	0.0716	0.1178	0.0833	0.7355
G-Softmax I, $\tau = 20$	0.1252	0.0869	0.1224	0.0975	0.0691	0.2500	0.7889
G-Softmax II, $\tau_1 = 1$, $\tau_2 = 20$	0.1066	0.1077	0.0560	0.1001	0.0506	0.2187	0.7624
G-Softmax II, $\tau_1 = 5$, $\tau_2 = 20$	0.1219	0.0881	0.0899	0.1110	0.0922	0.2282	0.7612
G-Softmax II, $\tau_1 = 30$, $\tau_2 = 20$	0.1087	0.0976	0.0816	0.1107	0.0733	0.1801	0.7567
HM-Attention, $m = 1$, $h_p = 0.5$	0.0919	0.1311	0.0624	0.0763	0.0954	0.0954	0.7282
HM-Attention, $m = 2$, $h_p = 0.5$	0.1032	0.1391	0.0753	0.0862	0.0823	0.1331	0.7284
HM-Attention, $m = 3$, $h_p = 0.5$	0.1047	0.1589	0.0771	0.0837	0.0724	0.1314	0.7299
HM-Attention, $m = 4$, $h_p = 0.5$	0.0958	0.1313	0.0669	0.0914	0.0591	0.1305	0.7325
HM-Attention, $m = 1$, $h_p = 0.1$	0.0942	0.1265	0.0677	0.0762	0.0900	0.1107	0.7260
HM-Attention, $m = 1$, $h_p = 0.2$	0.0938	0.1178	0.0700	0.0835	0.0837	0.1138	0.7278
HM-Attention, $m = 2$, $h_p = 0.1$	0.1006	0.1156	0.0641	0.1052	0.0893	0.1289	0.7332
HM-Attention, $m = 2$, $h_p = 0.2$	0.1114	0.0793	0.0834	0.0978	0.0513	0.2453	0.7843
HM-Attention, $n_{\text{heads}} = 16$,$m = 4$, $h_p = 0.5$	0.1002	0.1178	0.0725	0.0989	0.0902	0.1217	0.7301
HM-Attention, $n_{\text{heads}} = 16$,$m = 6$, $h_p = 0.5$	0.0985	0.1036	0.1024	0.0824	0.0735	0.1304	0.7354
HM-Attention, $n_{\text{heads}} = 16$,$m = 4$, $h_p = 0.1$	0.2038	0.0978	0.1788	0.1746	0.2060	0.3615	0.8069
HM-Attention, $n_{\text{heads}} = 16$,$m = 6$, $h_p = 0.1$	0.0879	0.1386	0.0571	0.0752	0.0616	0.1070	0.7310

Table 6.2 ECE and NLL Scores for CheXpert ID-Evaluation

All experiments were trained for 100 epochs during Self-Supervised-Training. Increasing epochs to 300 did not benefit the performance of the baseline model to a high degree, which is why we did not further explore an increasement of epochs with the other experiments. Deep Ensemble scores the highest on AUROC as expected. Hierarchical-Masked-Attention performs almost on par on AUROC with Deep Ensemble, while surpassing MC-Dropout and Gumbel-Softmax Attention, and improves ECE at the same time. MC-Dropout also performs strongly regarding ECE. NLL for Hierarchical-Masked Attention is slightly below the competitors, but still in a close range.

We can observe a beneficial effect of HMA and MC-Dropout on ECE for many experiments. Gumbel Softmax performs comparatively weak, especially when using Stage II and high τ-Values.

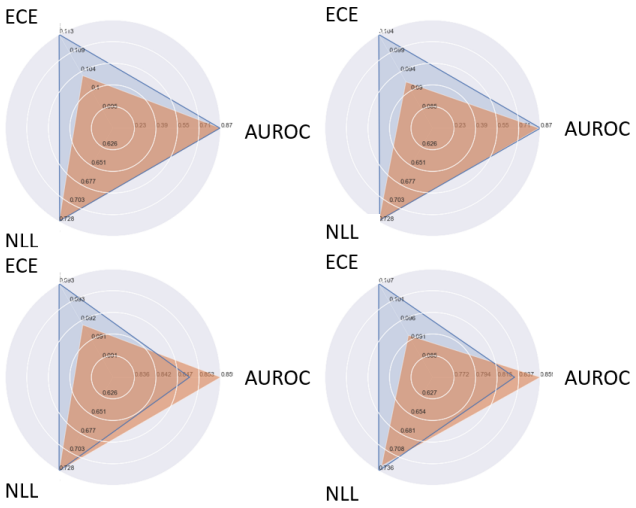


Figure 17 | Comparison of HMA (Red) to the best Competitors (Blue). (Top Left) Baseline (Top Right) Ensemble (Bottom Left) Gumbel-Softmax (Bottom Right) MC-Dropout. For AUROC, closer to the Edges is better, for ECE and NLL closer to the Center is better.

6.1.2 OOD-Evaluation

Experiment	Average	Cardiomegaly	Edema	Consolidation	Atelectasis	Pleural Effusion
Baseline, Seed 1	0.7356	0.7276	0.8460	0.8394	0.6511	0.6141
Baseline, Seed 2	0.7296	0.7344	0.8563	0.8819	0.5851	0.5905
Baseline, Seed 3	0.7391	0.7231	0.8638	0.8878	0.6208	0.5999
Baseline, 300 Epochs	0.7462	0.7745	0.8247	0.8229	0.7133	0.5956
Ensemble (100,100,100)	0.7291	0.7116	0.8465	0.8595	0.6185	0.6093
Ensemble (300,100,100)	0.7469	0.7497	0.8251	0.9032	0.6783	0.5780
Vit Base	0.7506	0.7085	0.8749	0.8158	0.7211	0.6325
MC-Dropout (0.1)	0.7335	0.6949	0.8522	0.8194	0.7125	0.5883
MC-Dropout (0.15)	0.7263	0.7269	0.8563	0.8678	0.6270	0.5536
MC-Dropout (0.2)	0.6496	0.5299	0.7499	0.7875	0.6123	0.5686
G-Softmax I, $\tau = 1$	0.7509	0.7959	0.8843	0.7237	0.7288	0.6218
G-Softmax I, $\tau = 3$	0.7047	0.7622	0.8411	0.6139	0.6465	0.6595
G-Softmax I, $\tau = 5$	0.7434	0.7514	0.8505	0.8335	0.6807	0.6012
G-Softmax I, $\tau = 20$	0.6067	0.5112	0.6702	0.6741	0.6053	0.5725
G-Softmax II, $\tau_1 = 1$, $\tau_2 = 20$	0.6943	0.6898	0.7753	0.6564	0.7420	0.6081
G-Softmax II, $\tau_1 = 5$, $\tau_2 = 20$	0.6753	0.6918	0.7454	0.7438	0.6138	0.5815
G-Softmax II, $\tau_1 = 30$, $\tau_2 = 20$	0.6549	0.6558	0.7107	0.6718	0.5742	0.6621
HM-Attention, $m = 1$, $h_p = 0.5$	0.7550	0.6891	0.8598	0.8654	0.7040	0.6569
HM-Attention, $m = 2$, $h_p = 0.5$	0.7329	0.7286	0.8336	0.8312	0.6706	0.6008
HM-Attention, $m = 3$, $h_p = 0.5$	0.7415	0.7435	0.8607	0.8323	0.6309	0.6402
HM-Attention, $m = 4$, $h_p = 0.5$	0.7331	0.7333	0.8433	0.8383	0.6193	0.6312
HM-Attention, $m = 1$, $h_p = 0.1$	0.7258	0.7429	0.8545	0.8371	0.6162	0.5785
HM-Attention, $m = 1$, $h_p = 0.2$	0.7387	0.7398	0.8353	0.8867	0.6317	0.5999
HM-Attention, $m = 2$, $h_p = 0.1$	0.7316	0.7092	0.8327	0.8489	0.6744	0.5926
HM-Attention, $m = 2$, $h_p = 0.2$	0.6158	0.5529	0.6907	0.7426	0.5882	0.5047
HM-Attention, $n_{\text{heads}} = 16$,$m = 4$, $h_p = 0.5$	0.7294	0.6901	0.8465	0.8040	0.6892	0.6171
HM-Attention, $n_{\text{heads}} = 16$,$m = 6$, $h_p = 0.5$	0.7044	0.5939	0.8465	0.7910	0.6861	0.6046
HM-Attention, $n_{\text{heads}} = 16$,$m = 4$, $h_p = 0.1$	0.7446	0.7337	0.8696	0.8194	0.6861	0.6145
HM-Attention, $n_{\text{heads}} = 16$,$m = 6$, $h_p = 0.1$	0.6743	0.5731	0.7775	0.9020	0.5921	0.5270

Table 6.3 AUROC-Scores for CheXpert OOD-Evaluation

Hierarchical-Masked-Attention yields the best performance on OOD-evaluation, surpassing Gumbel-Softmax-Attention and ViT-Base.

6.1.3 Semi-Supervised-Evaluation

Experiment	AUROC 0.1	ECE 0.1	NLL 0.1	AUROC 0.01	ECE 0.01	NLL 0.01
Baseline, Seed 1	0.8015	0.1074	0.7395	0.7647	0.0987	0.7457
Baseline, Seed 2	0.8369	0.1061	0.7241	0.8023	0.1147	0.7363
Baseline, Seed 3	0.8410	0.1068	0.7237	0.7826	0.1024	0.7420
Baseline, 300 Epochs	0.8381	0.0953	0.7178	0.7780	0.0958	0.7245
Ensemble (300,100,100)	0.8388	0.1034	0.7220	0.7836	0.1052	0.7386
Vit Base	0.8328	0.1016	0.7219	0.7727	0.0978	0.7395
MC-Dropout (0.1)	0.8376	0.1096	0.7257	0.7894	0.1078	0.7357
MC-Dropout (0.15)	0.8354	0.1051	0.7270	0.7999	0.1150	0.7416
G-Softmax I, $\tau = 1$	0.8269	0.1113	0.7307	0.7882	0.1117	0.7405
G-Softmax I, $\tau = 3$	0.8172	0.1088	0.7291	0.7333	0.1017	0.7470
G-Softmax I, $\tau = 5$	0.8252	0.1049	0.7284	0.7637	0.1125	0.7449
HM-Attention, $m = 1$, $h_p = 0.5$	0.8432	0.1108	0.7235	0.7831	0.1102	0.7439
HM-Attention, $m = 2$, $h_p = 0.5$	0.8327	0.1045	0.7274	0.7893	0.1094	0.7436
HM-Attention, $m = 3$, $h_p = 0.5$	0.8439	0.1051	0.7245	0.7864	0.0945	0.7393
HM-Attention, $m = 4$, $h_p = 0.5$	0.8360	0.1051	0.7252	0.7870	0.1001	0.7411
HM-Attention, $m = 1$, $h_p = 0.1$	0.8468	0.1091	0.7224	0.7953	0.1089	0.7377
HM-Attention, $m = 1$, $h_p = 0.2$	0.8352	0.1020	0.7250	0.7711	0.1165	0.7454
HM-Attention, $m = 2$, $h_p = 0.1$	0.8182	0.0982	0.7329	0.7589	0.0989	0.7446
HM-Attention, $m = 2$, $h_p = 0.2$	0.8354	0.1043	0.7232	0.7824	0.0979	0.7391
HM-Attention, $n_{\text{heads}} = 16$,$m = 4$, $h_p = 0.5$	0.8215	0.1144	0.7385	0.7306	0.1033	0.7578
HM-Attention, $n_{\text{heads}} = 16$,$m = 6$, $h_p = 0.5$	0.8040	0.1117	0.7423	0.7103	0.1075	0.7596
HM-Attention, $n_{\text{heads}} = 16$,$m = 4$, $h_p = 0.1$	0.8232	0.1130	0.7342	0.7296	0.1060	0.7537
HM-Attention, $n_{\text{heads}} = 16$,$m = 6$, $h_p = 0.1$	0.8209	0.0953	0.7221	0.7575	0.0897	0.7386

Table 6.4 Average AUROC Scores for CheXpert 0.1 and 0.01 Semi-Linear-Evaluation

All Top 3 AUROC-Scores on 0.1 Semi-Linear-Evaluation are achieved by Hierarchical-Masked-Attention, as well as two of the three top scores for ECE and NLL. For 0.1 Semi-Linear-Evaluation it looses two of its top positions for AUROC, and all for NLL. ECE Top 3 still includes two HMA-Experiments.

6.1.4 Transfer Learning

Experiment	Average	Cardiomegaly	Edema	Consolidation	Atelectasis	Pleural Effusion
Baseline, Seed 1	0.8885	0.8337	0.9359	0.9619	0.8091	0.9019

Table 6.5 AUROC Scores for CheXpert Transfer-Learning

Transfer-Learning yields the best AUROC score for CheXpert, whereas ECE is weaker than with Hierarchical-Masked-Attention and MC-Dropout:

Experiment	Average	Cardiomegaly	Edema	Consolidation	Atelectasis	Pleural Effusion	NLL
Baseline, Seed 1	0.1013	0.1560	0.0610	0.0744	0.0828	0.1323	0.7153

Table 6.6 ECE and NLL Scores for CheXpert Transfer-Learning

6.2 Eyepacs

EyePACS needed more training time during Self-Supervised-Pretraining than CheXpert in order for the ViT to learn representations that perform well in evaluation. A training time of 100 epochs failed to achieve suitable representations for all experiments. At 300 epochs Baseline, MC-Dropout and HMA started to deliver good results. Gumbel-Softmax started performing well at 800 epochs. We further explored an increasement of training time to 2400 epochs for Baseline and HMA.

6.2.1 In-Distribution Evaluation

Experiment	Binary Accuracy	ECE	NLL
Baseline, 300 Epochs	0.8181	0.0091	0.4405
MC-Dropout (0.15), 300 Epochs	0.8167	0.0072	0.2187
HM-Attention, $m = 1$, $h_p = 1.5$, 300 Epochs	0.8255	0.0082	0.3335
HM-Attention, $m = 1$, $h_p = 0.75$, 300 Epochs	0.8217	0.0087	0.3618
HM-Attention, $m = 1$, $h_p = 0.35$, 300 Epochs	0.8249	0.0119	0.3612
HM-Attention, $m = 2$, $h_p = 1.5$, 300 Epochs	0.8253	0.0087	0.3370
Baseline, 800 Epochs	0.8368	0.0098	0.4262
G-Softmax I, $\tau = 1$, 800 Epochs	0.8331	0.0086	0.2245
HM-Attention, $m = 1$, $h_p = 4$, 800 Epochs	0.8467	0.0111	0.2037
Baseline, 2400 Epochs	0.8643	0.0103	0.0646
HM-Attention, $m = 1$, $h_p = 12$, 2400 Epochs	0.8644	0.0109	0.0859

Table 6.7 Binary Accuracy, ECE and NLL Scores for EyePACS ID-Evaluation

300 Training Epochs. Hierarchical-Masked-Attention yields the best Binary Accuracy. All four HMA-Experiments surpass Baseline and MC-Dropout, as well as Gumbel-Softmax which was not learning good representations at 300 epochs. MC-Dropout delivers the best ECE and NLL, but Hierachical-Masked-Attention still sees an improvement compared to the Baseline-Model.

800 Training Epochs. HMA again results in the best Binary Accuracy-Score as well as the best NLL, however Gumbel-Softmax beats it to ECE.

2400 Training Epoch. Binary Accuracy and ECE are fairly close together for Hierarchical-Masked-Attention and the Baseline-Model, though the Baseline-Model achieves a better NLL.

Overall, we can see that Hierarchical-Masked-Attention increases performance on Accuracy and at the same time can yield a better Model Calibration. With enough training time, when the models likely have reached their capacity, performance reaches comparable levels for Baseline and HMA.

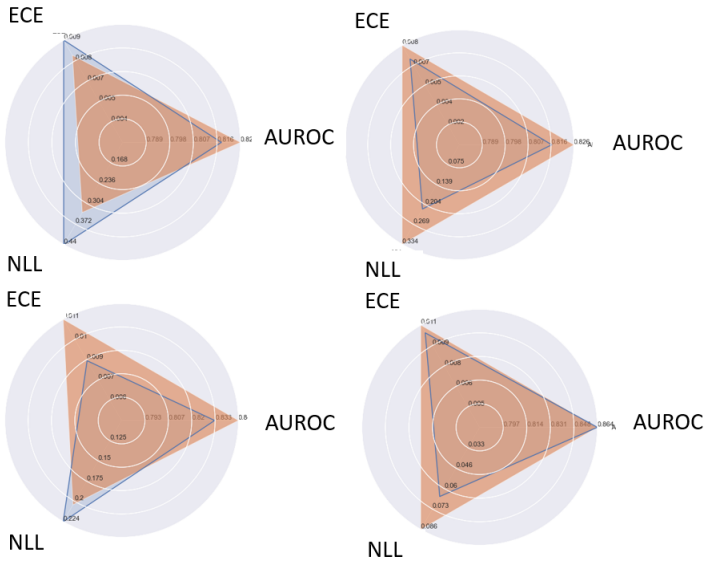


Figure 18 | Comparison of HMA (Red) to the best Competitors (Blue) on equal training time. (Top Left) Baseline, 300 (Top Right) MC-Dropout, 300 (Bottom Left) Gumbel-Softmax, 800 (Bottom Right) Baseline, 2400. For Accuracy, closer to the Edges is better, for ECE and NLL closer to the Center is better.

6.2.2 OOD-Evaluation

Experiment	Binary Accuracy
Baseline, 300 Epochs	0.7608
MC-Dropout (0.15), 300 Epochs	0.7600
HM-Attention, $m = 1$, $h_p = 1.5$, 300 Epochs	0.8282
HM-Attention, $m = 1$, $h_p = 0.75$, 300 Epochs	0.7783
HM-Attention, $m = 1$, $h_p = 0.35$, 300 Epochs	0.7769
HM-Attention, $m = 2$, $h_p = 1.5$, 300 Epochs	0.8069
Baseline, 800 Epochs	0.8640
G-Softmax I, $\tau = 1$, 800 Epochs	0.8356
HM-Attention, $m = 1$, $h_p = 4$, 800 Epochs	0.8779
Baseline, 2400 Epochs	0.8987
HM-Attention, $m = 1$, $h_p = 12$, 2400 Epochs	0.8949

Table 6.8 Binary Accuracy-Score for EyePACS OOD-Evaluation

Hierachical-Masked-Attention again achieves the best performance in OOD-Evaluation for 300 and 800 epochs training time. For 2400 epochs, performance is almost on-par.

6.2.3 Semi-Supervised-Evaluation

Experiment	Binary Accuracy 0.1	Binary Accuracy 0.01
MC-Dropout (0.15), 300 Epochs	0.8119	—-
HM-Attention, $m = 1$, $h_p = 1.5$, 300 Epochs	0.8167	—-
HM-Attention, $m = 1$, $h_p = 0.75$, 300 Epochs	0.8120	—-
Baseline, 800 Epochs	0.8343	—-
G-Softmax I, $\tau = 1$, 800 Epochs	0.8203	—-
HM-Attention, $m = 1$, $h_p = 4$, 800 Epochs	0.8335	0.8144
Baseline, 2400 Epochs	0.8531	0.8502
HM-Attention, $m = 1$, $h_p = 12$, 2400 Epochs	0.8538	0.8220

Table 6.9 Binary Accuracy Score for EyePACS 0.1 and 0.01 Semi-Linear-Evaluation

Hierarchical-Masked-Attention achieves the highest performance for 800 Epochs training time. For 300 and 2400, it is again almost on-par with Baseline. Experiments that performed poorly were omitted.

6.2.4 Transfer Learning

Experiment	Binary Accuracy	ECE	NLL
Baseline, 800 Epochs	0.8647	0.0100	0.0477
Baseline, 2400 Epochs	0.8682	0.0138	0.0545

Table 6.10 Binary Accuracy, ECE and NLL-Scores for EyePACS Transfer Learning

Notably, initializing the network with weights pretrained on ImageNet before Self-Supervised-Training results in just a marginal improvement of performance.

6.2.5 Visualization of Attention Heads

DINO has implemented a function to plot Self-Attention maps that can be used to visualize the information contained in different heads about the segmentation of an input image. An excerpt of these Attention maps layered over their original input images is given below for six sampled of each Data Set. We can see that the attention heads learn to focus on different shapes and structures or small details.

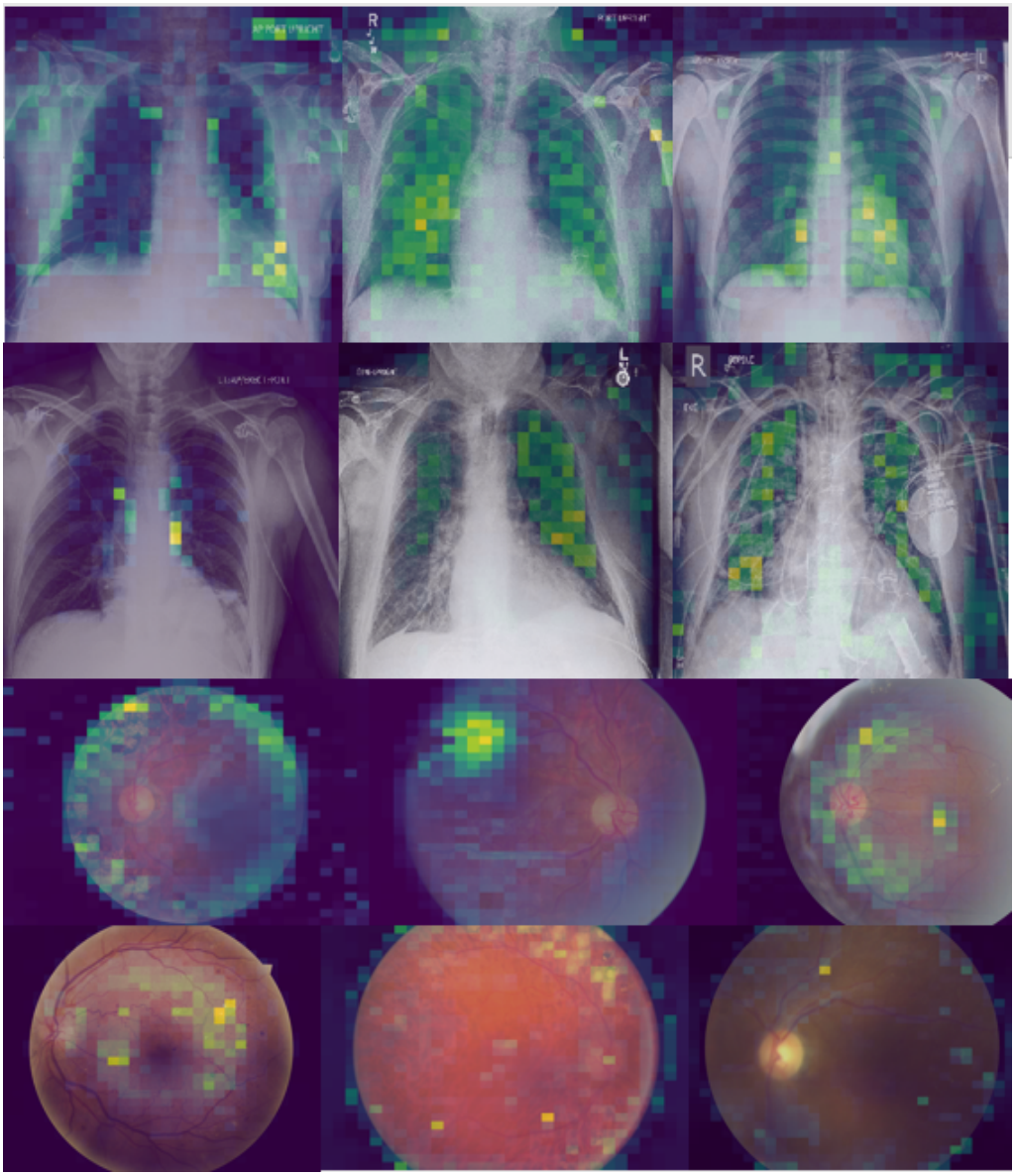


Figure 19 | Attention Visualization. (Row 1 & 2) Samples from CheXpert Validation Set (Row 3 & 4) Samples from EyePACS Training Set.

7. Conclusion and Future Work

Our experiments have shown that our newly introduced method, Hierarchical-Masked-Attention, is suited to improve performance on Accuracy Measures as well as Model Calibration for Data Sets from different medical areas and can be applied effectively in a Self-Supervised Learning Setting. Our method does not only perform well in ID-Evaluation (Linear Evaluation as well as Semi-Supervised Linear Evaluation), but also in OOD-Evaluation, and could therefore help to tackle problems of distribution shifts, which occur often in medical settings. Beyond that, it can boost training efficiency compared to other methods (see training time needed to learn useful representations of HMA vs. Gumbel Softmax in 6.2)

As Hierachical-Masked-Attention introduces two new Hyperparameters, future work could investigate further on their effect, e.g. via a more granular experimentation on the thinning parameter h_p or an ablation study of the threshold parameter T. A study of the influence of DINO hyper parameters to our method might also be beneficial. Furthermore, an extension to more medical fields (e.g. Dermatoscope, Brain MRA or Breast Ultrasound) could also be of interest to further investigate on generalization potential. As our method introduces stochasticity to the ViT-Architecture, it would also seem promising to incorporate techniques of uncertainty awareness or uncertainty quantification.

As Generalization to ID and OOD-settings as well as Data-efficient training remain unsolved problems in the field of medical imaging analysis, we hope that this work can provide a small contribution in tackling these challenges or for future research to build upon.

Acknowledgements and Reproducibility

The code basis used for all experiments and the implementation of our method was the officialDINO-Repository. We make our extensions publicly available in the HMA-Repository. All data sets are publicly available as well, and therefore our results are reproducible.

I'd like to thank the supervisor of this masters thesis, Dr. Mina Rezaei, for her continuous support throughout the whole process of work, my mother, Susanne Baur, for giving me the opportunity to pursue this degree, and my partner, Deborah Epstein, for facing all challenges beyond medical imgaging analysis together.

List of Abbreviations

Expression	Representing
AI	Artificial Intelligence
AUROC	Area Under the Receiver Operating Characteristic Curve
DL	Deep Learning
ID	In Distribution
ECE	Expected Calibration Error
HMA	Hierchical-Masked-Attention
MLP	Multi-Layer-Perceptron
NLL	Negative Log-Likelihood
OOD	Out of Distribution
PLEX	Pretrained Large Model Extensions
sota	state-of-the-art
SNGP	Spectral-normalized Neural Gaussian Process
SSL	Self-Supervised-Learning
w.r.t	with respect to
ViT	Vision Transformer

List of Figures

List of Tables

Bibliography

[1] M. Abdar, F. Pourpanah, S. Hussain, D. Rezazadegan, L. Liu, M. Ghavamzadeh, P. Fieguth, X. Cao, A. Khosravi, U. R. Acharya, et al. A review of uncertainty quantification in deep learning: Techniques, applications and challenges. *Information Fusion*, 76:243–297, 2021.

[2] S. Azizi, L. Culp, J. Freyberg, B. Mustafa, S. Baur, S. Kornblith, T. Chen, P. MacWilliams, S. S. Mahdavi, E. Wulczyn, et al. Robust and efficient medical imaging with self-supervision. *arXiv preprint arXiv:2205.09723*, 2022.

[3] J. Baan. A comprehensive introduction to bayesian deep learning. `https://jorisbaan.nl/2021/03/02/introduction-to-bayesian-deep-learning.html#Basics`. Accessed: 2023-01-18.

[4] K. Bera, K. A. Schalper, D. L. Rimm, V. Velcheti, and A. Madabhushi. Artificial intelligence in digital pathology—new tools for diagnosis and precision oncology. *Nature reviews Clinical oncology*, 16(11):703–715, 2019.

[5] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 9650–9660, October 2021.

[6] N. C. Center. Chestx-ray14. `https://paperswithcode.com/dataset/chestx-ray14`. Accessed: 2023-02-08.

[7] R. Clarida, J. Galı, and M. Gertler. A simple framework for international monetary policy analysis. *Journal of monetary economics*, 49(5):879–904, 2002.

[8] J. Condon, L. Oakden-Rayner, K. Hall, M. Reintals, A. Holmes, G. Carneiro, and L. Palmer. Replication of an open-access deep learning system for screening mammography: Reduced performance mitigated by retraining on local data. *medRxiv*, 2021.

[9] J. De Fauw, J. R. Ledsam, B. Romera-Paredes, S. Nikolov, N. Tomasev, S. Blackwell, H. Askham, X. Glorot, B. O'Donoghue, D. Visentin, et al. Clinically applicable deep learning for diagnosis and referral in retinal disease. *Nature medicine*, 24(9):1342–1350, 2018.

[10] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[11] A. Esteva, B. Kuprel, R. A. Novoa, J. Ko, S. M. Swetter, H. M. Blau, and S. Thrun. Dermatologist-level classification of skin cancer with deep neural networks. *nature*, 542(7639):115–118, 2017.

[12] S. Finlayson, A. Subbaswamy, K. Singh, J. Bowers, A. Kupke, J. Zittrain, I. Kohane, and S. Saria. The clinician and dataset shift in artificial intelligence. *New England Journal of Medicine*, 385:283–286, 07 2021.

[13] C. H. Foundation. Diabetic retinopathy detection. `https://www.kaggle.com/competitions/diabetic-retinopathy-detection/overview`. Accessed: 2023-02-08.

[14] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059. PMLR, 2016.

[15] E. J. Gumbel. *Statistical theory of extreme values and some practical applications: a series of lectures*, volume 33. US Government Printing Office, 1954.

[16] J. Z. HaoChen, C. Wei, A. Kumar, and T. Ma. Beyond separability: Analyzing the linear transferability of contrastive representations to related subpopulations. *arXiv preprint arXiv:2204.02683*, 2022.

[17] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[18] G. Hinton, O. Vinyals, and J. Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[19] A. E. Hospital. Aptos 2019 blindness detection. `https://www.kaggle.com/competitions/aptos2019-blindness-detection/overview`. Accessed: 2023-02-08.

[20] Z. Huang, H. Lam, and H. Zhang. Quantifying epistemic uncertainty in deep learning. *arXiv preprint arXiv:2110.12122*, 2021.

[21] S. inlayson, A. Subbaswamy, K. Singh, J. Bowers, A. Kupke, J. Zittrain, I. Kohane, and S. Saria. Underspecification presents challenges for credibility in modern machine learning, 2020.

[22] J. Irvin, P. Rajpurkar, M. Ko, Y. Yu, S. Ciurea-Ilcus, C. Chute, H. Marklund, B. Haghgoo, R. Ball, K. Shpanskaya, et al. Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 590–597, 2019.

[23] A. Kendall and Y. Gal. What uncertainties do we need in bayesian deep learning for computer vision? *Advances in neural information processing systems*, 30, 2017.

[24] B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. *Advances in neural information processing systems*, 30, 2017.

[25] J. Liu, Z. Lin, S. Padhy, D. Tran, T. Bedrax Weiss, and B. Lakshminarayanan. Simple and principled uncertainty estimation with deterministic deep learning via distance awareness. *Advances in Neural Information Processing Systems*, 33:7498–7512, 2020.

[26] P. Liu, W. Yuan, J. Fu, Z. Jiang, H. Hayashi, and G. Neubig. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. *ACM Computing Surveys*, 55(9):1–35, 2023.

[27] I. Loshchilov and F. Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016.

[28] I. Loshchilov and F. Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[29] S. M. McKinney, M. Sieniek, V. Godbole, J. Godwin, N. Antropova, H. Ashrafian, T. Back, M. Chesus, G. S. Corrado, A. Darzi, et al. International evaluation of an ai system for breast cancer screening. *Nature*, 577(7788):89–94, 2020.

[30] J. Pei, C. Wang, and G. Szarvas. Transformer uncertainty estimation with hierarchical stochastic attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11147–11155, 2022.

[31] M. E. Sander, P. Ablin, M. Blondel, and G. Peyré. Sinkformers: Transformers with doubly stochastic attention. In *International Conference on Artificial Intelligence and Statistics*, pages 3515–3530. PMLR, 2022.

[32] J. C. Y. Seah, C. H. M. Tang, Q. D. Buchlak, X. G. Holt, J. B. Wardman, A. Aimoldin, N. Esmaili, H. Ahmad, H. Pham, J. F. Lambert, B. Hachey, S. J. F. Hogg, B. P. Johnston, C. Bennett, L. Oakden-Rayner, P. Brotchie, and C. M. Jones. Effect of a comprehensive deep-learning model on the accuracy of chest x-ray interpretation by radiologists: a retrospective, multireader multicase study. *The Lancet Digital Health*, 3(8):e496–e506, 2021.

[33] R. Sinkhorn. A relationship between arbitrary positive matrices and doubly stochastic matrices. *The annals of mathematical statistics*, 35(2):876–879, 1964.

[34] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[35] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pages 10347–10357. PMLR, 2021.

[36] D. Tran, J. Liu, M. W. Dusenberry, D. Phan, M. Collier, J. Ren, K. Han, Z. Wang, Z. Mariet, H. Hu, et al. Plex: Towards reliability using pretrained large model extensions. *arXiv preprint arXiv:2207.07411*, 2022.

[37] O. F. Tuna, F. O. Catak, and M. T. Eskil. Exploiting epistemic uncertainty of the deep learning models to generate adversarial samples. *Multimedia Tools and Applications*, 81(8):11479–11500, 2022.

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

[39] A. G. Wilson and P. Izmailov. Bayesian deep learning and a probabilistic perspective of generalization. *Advances in neural information processing systems*, 33:4697–4708, 2020.

[40] J. R. Zech, M. A. Badgeley, M. Liu, A. B. Costa, J. J. Titano, and E. K. Oermann. Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: a cross-sectional study. *PLoS medicine*, 15(11):e1002683, 2018.

[41] H. Zhang, N. Dullerud, L. Seyyed-Kalantari, Q. Morris, S. Joshi, and M. Ghassemi. An empirical framework for domain generalization in clinical settings. pages 279–290, 04 2021.

[42] W. Zhou, T. Ge, K. Xu, F. Wei, and M. Zhou. Scheduled drophead: A regularization method for transformer models. *arXiv preprint arXiv:2004.13342*, 2020.