Segmentation Consistency Training: Out-of-Distribution Generalization for Medical Image Segmentation

Anonymous Author(s)
Affiliation
Address
email

Abstract

Generalizability is seen as one of the major challenges in deep learning especially in domains such as medicine where a change of the hospital can lead to a complete failure of a model. To tackle this we introduce **Consistency Training**, a training procedure based on maximizing models' consistency across augmented and unaugmented data in order to facilitate better out-of-distribution generalization. To this end, a novel segmentation loss function is developed based on the Jaccard loss, which in addition considers the difference between pairs of augmented and unaugmented predictions and labels. Our method is shown to outperform conventional data augmentation on a popular medical task, namely, polyp-segmentation. We highlight continued development of this training paradigm as a promising direction of further research towards increasing the generalizability of deep learning models.

1 Introduction

The last decade or so has seen a veritable revolution in Artificial Intelligence (AI). This has in large part been spearheaded by advancements in Deep Learning, the remarkable performance of which has rendered more conventional approaches practically obsolete. Recent work has however highlighted that Deep Neural Networks (DNNs) are highly prone to exhibiting significant reductions in performance when deployed in practical settings, in spite of the fact that they readily exhibit high performance when evaluated on previously unseen subsets of the training data [8, 10, 12, 16]. This is referred to as *generalization failure*.

Recent analyses attribute generalization failure to a structural misalignment between the features that a given model learns through Empirical Risk Minimization (ERM) and the causal structure which it ideally should encode [3, 10, 17, 24]. Generally, this misalignment occurs as a result of the predictor learning spurious or otherwise causally unrepresentative features that nonetheless perform well within the training distribution. This is often referred to as *shortcut learning* [10] or the *Clever Hans effect* [19]. This behaviour is of course made evident as soon as the predictor is exposed to any form of distributional shift which breaks these shortcuts, at which point it will fail to generalize. These distributional shifts can range in magnitude, from common corruptions such as noise or blurs [13] or spatial transforms [9], to practically imperceptible perturbations, typically exemplified by adversarial attacks [5], or as will be shown in this work simply collecting data from different centers. ERM does not and cannot guarantee invariance to these sorts of distributional shifts, as it assumes that the distribution of the training data is Independent and Identically Distributed (IID) to the true distribution.

Closely related to shortcut learning is underspecification [8]. A machine learning pipeline can be considered underspecified when it can return any number of risk-equivalent predictors when evaluated

Submitted to 36th Conference on Neural Information Processing Systems (NeurIPS 2022). Do not distribute.

on an IID holdout set, dependent only on the random variables used within the training procedure - i.e dropout, weight initialization, and so on. Even with identical hyperparameters, a given training procedure can return any number of predictors each having learned different patterns within the dataset. One predictor may have learned one shortcut, another may have learned a different shortcut, and the next may actually have learned features that correspond to the causal structure it is intended to learn. With ERM, and in particular with In-Distribution (InD)-oriented evaluation procedures, these are all erroneously considered equivalent.

EndoCV2021 provided an opportunity to investigate generalization failure and means by which to counteract them in the context of detection- and segmentation of colorectal polyps via a competition [2]. Though several teams made good progress towards increasing generalizability, the proceedings highlighted that every submitted model nevertheless exhibited significant performance reductions on these unseen datasets. Moreover, though a multitude of methods and approaches were tested, many of which did indeed benefit generalizability, few of methods stood out as having the potential for significant further development.

To address these shortcomings, we introduce **Consistency Training**. We re-frame the problem of learning generalizable features into a matter of learning to *not* learn spurious features. This framework requires a *perturbation model*, which we in this work implement as simple data augmentation, and a differentiable quantity that represents the consistency of the predictions across perturbed and unperturbed inputs images, which we implement as *Segmentation Inconsistency Loss (SIL)*, a Jaccard-like loss function that quantifies the degree to which the segmentation probability maps exhibit unwarranted change after the input is perturbed. This loss function is then used in conjunction with a task-specific loss, in this work Jaccard loss. To increase the stability of the training routine, we also implement a dynamic weighing procedure for the two constituent components of the overall loss function. We show that Consistency Training increases generalization by a significant margin on all tested datasets when compared to conventional data augmentation.This framework is in other words a more performant alternative to data augmentation, which leads to increased generalization with no additional overhead asides from the added computational cost involved with the auxiliary loss term and the memory required to store augmented and un-augmented versions of each batch. We summarize our contributions as following:

-
-
-

2 Related Work

The development of consistency training was in large part informed by recent advances in the understanding of generalization failure. D'Amour et al. [8] perform a thorough analysis of generalization failure through multiple case studies and highlight the role of underspecification therein. Geirhos et al. [10] explore the idea of shortcut learning in a similar manner, and highlight the importance of learning causally related features. Schölkopf[24] discusses the importance of causality in machine learning and how it relates to generalization failure. Robey et al. [21] investigate the use of model-based training procedures towards mitigating generalization failure, in particular with regards to generative networks. Sandfort et al. [23] also use generative networks as data augmentation to improve generalization. Gokhale et al. [11] compare the use of multiple data modification methods on robustness and generalization and find that data augmentation improves generalizability by a significant margin. Finally, Hendrycks et Al. [14] leverage the use of a similar consistency-term in order to facilitate robustness to distributional shifts for the image-classification task.

In the context of polyp-segmentation, this work was motivated in large part by the findings in the proceedings of EndoCV2021 [2], which through the evaluation of submissions on multiple Out of Distribution (OOD) datasets highlighted the significance of generalization failure. The winning submission to EndoCV2021, submitted by Thambawita et Al. [26] leverages an ensemble-network in order to increase generalizability. Honga et Al. [15] also implement an ensemble-based model, which they show improves generalization.

3 Approach

3.1 Consistency Training Method

This section will introduce Consistency Training, a training procedure wherein the objective is to optimize for invariance to a set of various image transformations by quantifying the degree to which the model outputs inconsistent predictions when its input is subjected to some transformations. This is achieved by giving the model two images: one which is augmented, and on which is not. These inputs are then passed through the model, resulting in two segmentation masks. The difference between these two predictions is then computed, and compared to the difference (if any) between the augmented and unaugmented segmentation labels. This is then incorporated into the loss-function such that the discrepancy between the expected prediction change and actual prediction change is minimized. This is illustrated in 1. The next sections will cover the theoretical basis of this training procedure as well as the implementation of its constituent components.

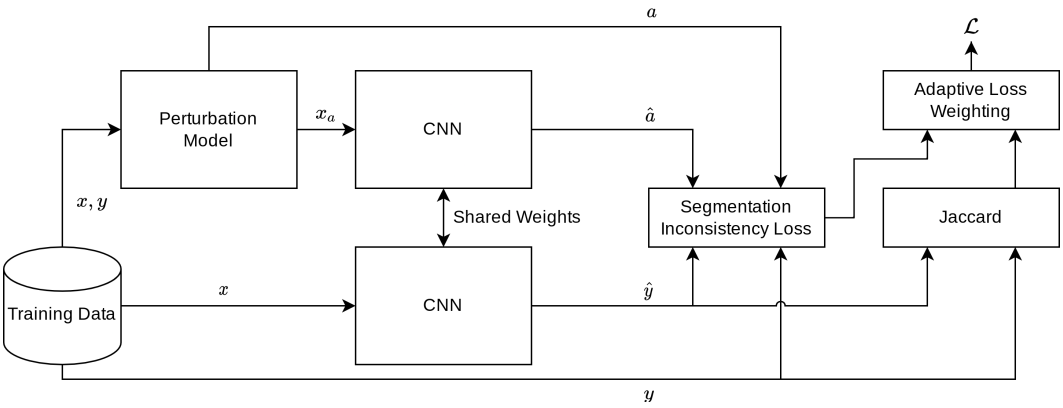

Figure 1: Consistency Training.

3.2 Quantifying Segmentation Consistency

Let $Y := \{y, \hat{y} := f(x)\}$ be the set consisting of the segmentation labels (masks) and predictions for the unperturbed samples, where $f(\cdot)$ as before denotes the model. Let $\epsilon(\cdot)$ be some perturbation function. Then, let $A := \{a := \epsilon(y), \hat{a} := f(\epsilon(x))\}$ be the set consisting of segmentation predictions and masks when the input is subjected to a perturbation. Segmentation consistency can then be quantified as:

$$\mathcal{C}(A, Y) = \frac{\sum\{y \cap a \cap \hat{y} \cap \hat{a}\}}{\sum\{y \cup a \cup \hat{y} \cup \hat{a}\}} \tag{1}$$

Equivalently, *inconsistency* can be quantified as:

$$\overline{\mathcal{C}}(A, Y) = \frac{1}{\sum\{y \cup a \cup \hat{y} \cup \hat{a}\}} \sum\{y \ominus \hat{y} \ominus a \ominus \hat{a}\} \tag{2}$$

These formulations are, of course, related by:

$$\mathcal{C}(A, Y) = 1 - \overline{\mathcal{C}}(A, Y)$$

In simple terms, this quantity corresponds to counting the number of pixels that change after the input is subjected to a perturbation - $\hat{a} \ominus \hat{y}$, but discounting those we expect to change, $a \ominus y$. This is shown in Figure 2.

Inconsistency as expressed in Equation (2) is not differentiable, and thus it cannot in its current state be used as a part of a loss function. This, naturally, limits the utility of the idea somewhat. Thus, a smooth extension of this metric is needed which can be achieved in much the same way as how the

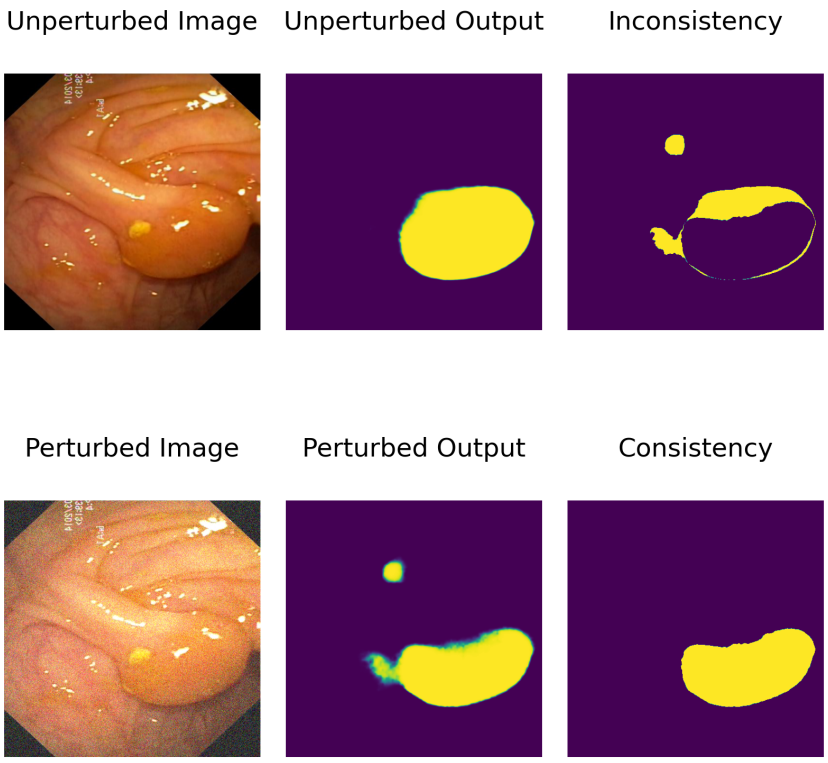

Figure 2: Examples of consistency and inconsistency calculation when the input is subjected to additive noise. The consistency for this sample is 0.68 and inconsistency 0.32, meaning that 64% of the pixels constitute consistent predictions across the two inputs

Jaccard loss can be derived from the Jaccard index - i.e by using differentiable versions of the set functions.

We can extend the definition of the symmetric difference to $\Theta(A, B) = A(1 - B) + B(1 - A)$. This, naturally, is equivalent to the standard symmetric difference if the values of A and B are binary. Similarly, the union operator can be extended as $\bigcup(A, B) = A + B - AB$, and the intersection operator as $\bigcap(A, B) = AB$. Like its binary equivalents, these operators maintain their associative and commutative properties. One can optimize for consistency by replacing the operators in Equation (2) with these functions, which in turn can be used as a loss function:

$$L_c(y, \hat{y}, a, \hat{a}) = \sum \frac{\Theta(y, \hat{y}, a, \hat{a})}{\bigcup(y, \hat{y}, a, \hat{a})} \tag{3}$$

This loss function will from this point be referred to as the SIL.

3.3 Incorporating Consistency into Training

Naturally, using SIL as a loss function on its own is not really useful since it only expresses inconsistency, and is to a large extent agnostic to whatever object it is trying to segment. For instance, if the perturbation being performed is simply additive noise, the loss is equally well minimized by predicting that every pixel is positive as it is by segmenting the polyps alone. Consequently, it has to be combined with a some conventional segmentation loss, for instance Jaccard loss. A simple way to do this would be to simply add them together and normalize, i.e:

$$L(Y, A) = \frac{1}{2}\big[L_{seg}(Y) + L_c(Y, A)\big]$$

Preliminary experiments showed that this, however, exhibited some degree of instability. The model would readily get stuck in local minima where its predictions were indeed consistent, but also consistently predicting artifacts. Examples of this can be found in the Appendix.

To mitigate this, it is possible to employ a weighing strategy. Instead of simply adding the respective losses together, one may weight the individual components adaptively according to the InD segmentation performance. This way, the model will learn to predict generally correct segmentations early in the training, then start weighing consistency and as a result generalization more and more as the model sees improvements to its segmentation performance:

$$L = (1 - IoU) \times L_{seg} + IoU \times L_c \tag{4}$$

Using this formulation, the model will start off trying to learn features that contribute to generally improved segmentation performance, then as segmentation performance improves start principally focusing on learning to be consistent. If the model starts veering into areas in the loss-landscape that constitute poor segmentation performance, it will self-correct by weighing the segmentation loss more. In the implementation used in this thesis, these Intersection over Union (IoU) weights were calculated on a per-batch basis such that the model can quickly adapt if either of the respective objectives exhibit a degradation in performance during training.

4 Experiments and Results

To ascertain the impact of Consistency Training, we trained ten predictors across four separate model architectures, namely DeepLabV3+, Unet, FPN, and TriUnet, using Consistency Training. These average generalizability of these predictors were then compared to the generalizability of the same models trained with and without data augmentation by testing the resulting models on three OOD datasets.

4.1 Experimental Setup

Models. To evaluate the impact of Consistency Training sufficiently, it was tested across a range of different models. These models include DeepLabV3+ [7], Feature Pyramid Network (FPN) [20], UNet [22], and Tri-Unet [26].

The models were implemented in pytorch using the segmentation-models-pytorch library [27], using the library's default values. This includes initialization with Imagenet-pretrained weights.

Ten instances of each model were trained across each configuration in order to perform statistical analysis.

Datasets. Naturally, the best way to evaluate the generalizability of a given predictor is to test it directly on OOD data. Though this can to some extent be achieved by carefully designing stress-tests [8], a more straight-forward approach is to simply leverage existing OOD datasets. To this end, a number of polyp-segmentation datasets were selected. The names, sizes, resolutions and availabilities of these datasets is shown in Table 1. Samples images and masks from the datasets can be seen in Figure 3. Kvasir-SEG was selected as the training dataset, and partitioned into a 80/10/10 split as training/validation/test data.

Metrics We used two metrics to evaluate generalizability. To evaluate raw performance, we used IoU, which is defined as follows:

Table 1: Dataset Overview

Dataset	Resolution	Size	Availability
Kvasir-SEG [18]	Variable	1000	Public
Etis-LaribDB [25]	1255x966	196	Public
CVC-ClinicDB [4]	388x288	612	Public
EndoCV2020 [1]	Variable	127	Request

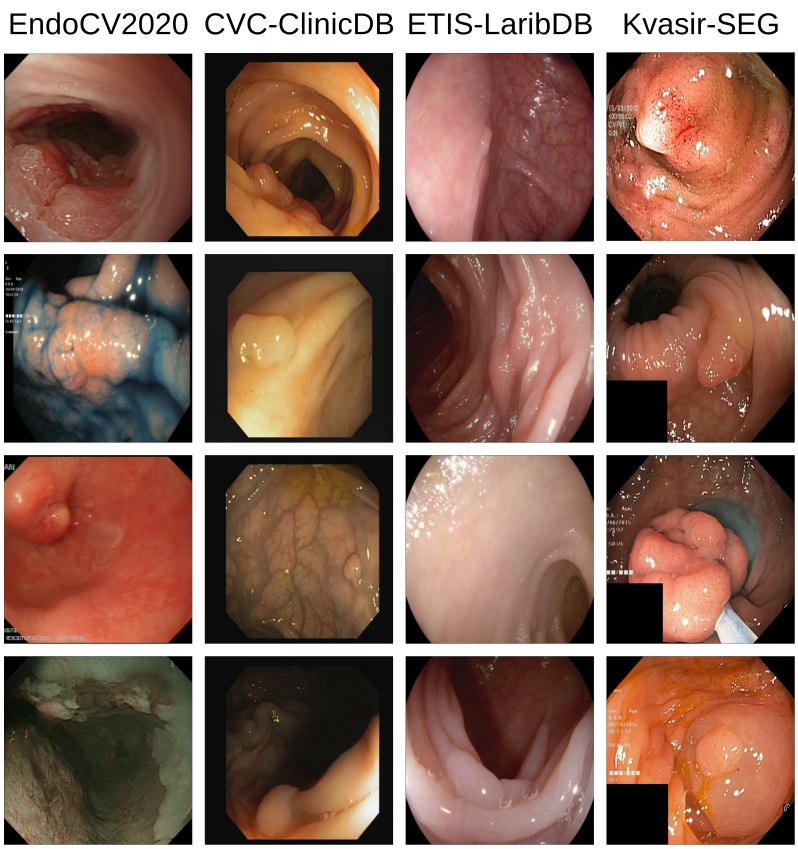

Figure 3: Sample images from the datasets.

$$IoU(y, \hat{y}) = \frac{\sum\{y = \hat{y}\}}{\sum\{y = 1\} \cup \{\hat{y} = 1\}\}}$$

Measuring the average IoU scores across all the aforementioned datasets, naturally, provide an indication of the generalizability of the given predictor. Though it is of course impossible to account for all distributional shifts that may occur in deployment, high degrees of generalization across multiple datasets should nevertheless indicate a sufficient level of generalization

We also analyzed the Coefficient of Standard Deviation (C.StD) of the IoUs across samples. This is defined as follows:

$$C.StD = \frac{1}{n\mu}\sqrt{\sum_{i}^{n}(\mu - x_i)^2} \tag{5}$$

Though the mean generalizability gap across these predictors is the primary indication of generalizability, this variability is also a salient factor to consider., as it serves to quantify the degree to which a given pipeline is underspecified. The more underspecified a pipeline is, the higher the variability of the performance and the higher the C.StD of the IoUs.

4.1.1 Implementation details

All experiments where conduced using Nvidia Tesla-V100 GPUs on the eX3 computing infrastructure offered by Simula Research Laboratory. The experiments were implemented in Python 3.8 using PyTorch and segmentation-models-pytorch [27]. The source code as well as all of the raw data is available at https://github.com/BirkTorpmannHagen/SegmentationConsistencyTraining.

The augmentation method used both for the baseline and as part of Consistency Training was implemented using the albumentations library [6], and consisted of the following transformations: RandomRotate90(), GaussNoise(), ImageCompression(), OpticalDistortion() and ColorJitter(). For the regular augmentation baseline, the augmentation probability was set to 0.5, in which case all of the aforementioned transformations were applied.

The hyperparameters used when training the models are shown in Table 2.

Table 2: Hyperparameters

Component	Type	Hyperparameters
Dataloader	-	$batch\_size = 8$ train/val/test split $= 80/10/10$
Optimizer	Adam	$lr = 0.00001$
Scheduler	Cosine Annealing w/ Warm Restarts	$T_0 = 50$ $T_{mult} = 2$
Evaluation	Loss-based Early Stopping	$epochs = 300$

4.2 OOD Generalization

table 3 shows the mean IoUs for models trained with and without data augmentation, and models trained with Consistency Training. Across all experiments, Consistency Training performs as well as or better than data augmentation. When averaging across models, Consistency Training improves generalization by a statistically significant margin (p>0.99) on all OOD datasets over conventional augmentation. This is shown in Figure 4.This shows that Consistency Training can be considered a more generalizable alternative to data augmentation.

4.3 Consistency Training and Underspecification

Figure 5 shows that Consistency Training reduces the variability in performance over data augmentation on three of the four datasets used. This suggests that the constraints imposed by Consistency Training to some extent mitigates underspecification. It should be noted, however, that the low sample size means that the estimates for the C.StD values carry some uncertainty.

5 Discussion and Conclusion

In this paper, we introduced Segmentation Consistency Training, a novel training procedure for segmentation which explicitly optimizes for consistent behaviour when an input subjected to augmentation. We showed that this improves OOD generalization by a statistically significant amount across several models when compared to conventional data augmentation. Moreover, we show that Consistency Training mitigates underspecification to a greater extent than data augmentation by analyzing performance variability.

5.1 Limitations

The batch size was kept constant across all experiments performed in this paper. However, as it can be argued that since Consistency Training implicitly increases the batch size, the experiments should ideally be repeated across a range of batch sizes.

Table 3: Mean IoUs for training methods, precision truncated to 99% confidence. Consistency training entries with greater performance than conventional augmentation by a statistically significant margin (p>0.99) after an independent sample two-sided t-test for the given model and dataset are highlighted in bold.

Model	No Augmentation	Vanilla Augmentation	Consistency Training
		Kvasir-SEG	
DD-DeepLabV3+	0.829	0.848	0.852
DeepLab	0.822	0.850	0.852
FPN	0.822	0.853	0.852
TriUnet	0.817	0.841	0.845
Unet	0.828	0.851	0.851
		Etis-LaribDB	
DD-DeepLabV3+	0.408	0.460	0.482
DeepLab	0.417	0.472	**0.505**
FPN	0.404	0.440	**0.475**
TriUnet	0.309	0.410	0.434
Unet	0.403	0.447	**0.481**
		CVC-ClinicDB	
DD-DeepLabV3+	0.681	0.728	0.736
DeepLabV3+	0.684	0.733	0.740
FPN	0.675	0.715	**0.727**
TriUnet	0.623	0.684	0.696
Unet	0.679	0.717	**0.730**
		EndoCV2020	
DD-DeepLabV3+	0.596	0.668	0.668
DeepLab	0.608	0.676	0.676
FPN	0.600	0.662	0.673
TriUnet	0.577	0.667	0.684
Unet	0.598	0.660	**0.676**

Moreover, the experiments were only performed with one specific augmentation strategy. As it may be the case that the differences are less significant given a more highly developed augmentation strategy, repeating the experiment with a range of different augmentation strategies may be warranted.

Finally, a larger number of samples should ideally have been collected across a wider diversity of models architectures. Increasing the granularity of the findings by other means, for instance by using a greater number of OOD datasets or designing parameterized stress-tests may also be warranted in order to develop a more thorough understanding of the impact of our methods.

5.2 Future Work

We believe the Consistency Training framework exhibits great potential for further development, and that it may constitute a promising candidate towards mitigating generalization failure.

We plan to investigate a number of potential improvements of this framework. Consistency was for instance in this paper quantified as the symmetric difference between the expected change in the output due to augmentation and the actual change due to augmentation. This is largely agnostic to the augmentation being performed. However, it may be beneficial to take the nature of these augmentations into account. If the image is subjected to a 90 degree rotation, for instance, the prediction would following the notion of consistency as used in this work be considered perfectly consistent so long as the pixels corresponding to the polyps are rotated, and the incorrectly classified pixels remain unchanged. However, if the model instead learns to rotate all of the pixels - even those that are incorrectly classified - it may learn a more accurate representation of what constitutes

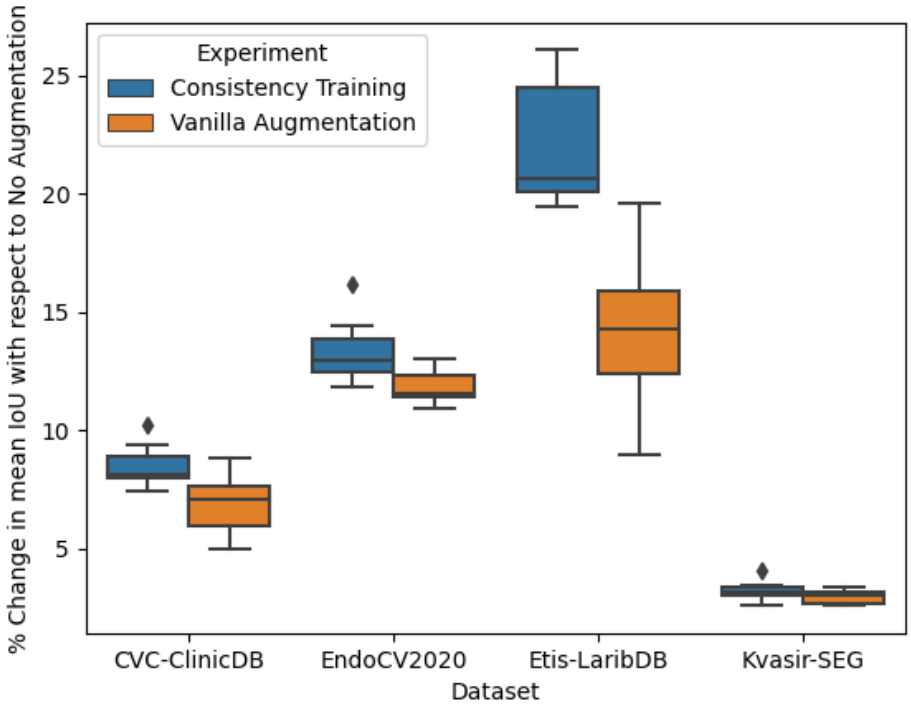

Figure 4: Improvements due Consistency Training and Data Augmentation as a percentage the mean IoU without augmentation across datasets

consistent behavior under rotation. I.e, instead of expressing inconsistency as:

$$\bar{\mathcal{C}} = y \ominus a \ominus \hat{y} \ominus \hat{a}$$

One can adjust the expected change term such that also incorrect predictions can be considered consistent so long as they change in accordance to the nature of the perturbation model $\epsilon(\cdot)$:

$$\bar{\mathcal{C}} = \hat{y} \ominus \hat{a} \ominus \hat{y} \ominus \epsilon(\hat{y})$$

This also has the advantage of being independent of the labels themselves. This may alleviate complications that may arise as a consequence of poor and/or incomplete labeling which would otherwise affect what the models learn to associate with consistent behaviour.

Further, one could investigate whether the consistency-training framework also can be implemented in the context of classification, object detection, or other applications of Deep Learning, and if similar improvements to generalizability can be shown in other domains.

References

[1] Sharib Ali et al., eds. *EndoCV2020: 2nd International Workshop and Challenge on Computer Vision in Endoscopy*. Vol. 2595. Iowa, USA: CEUR Workshop Proceedings, 2020. URL: http://ceur-ws.org/Vol-2595/.

[2] Sharib Ali et al., eds. *Proceedings of the 3rd International Workshop and Challenge on Computer Vision in Endoscopy (EndoCV 2021) co-located with with the 17th IEEE International Symposium on Biomedical Imaging (ISBI 2021)*.

[3] Martin Arjovsky et al. *Invariant Risk Minimization*. 2019. DOI: 10.48550/ARXIV.1907.02893. URL: https://arxiv.org/abs/1907.02893.

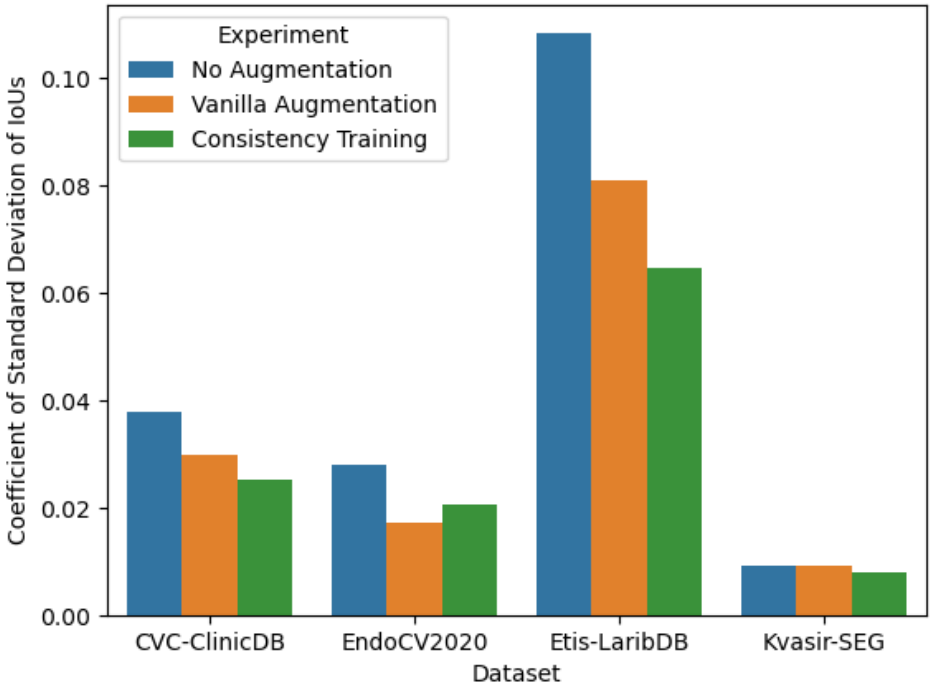

Figure 5: Models trained with Consistency Training exhibit lower predictor-wise performance variability than models trained without augmentation or with regular data augmentation

[4] Jorge Bernal et al. "WM-DOVA maps for accurate polyp highlighting in colonoscopy: Validation vs. saliency maps from physicians". In: *Computerized Medical Imaging and Graphics* 43 (2015), pp. 99–111. DOI: http://dx.doi.org/10.1016/j.compmedimag.2015.02.007. URL: https://polyp.grand-challenge.org/CVCClinicDB/.

[5] Battista Biggio et al. "Evasion Attacks against Machine Learning at Test Time". In: *Lecture Notes in Computer Science* (2013), 387–402. ISSN: 1611-3349. DOI: 10.1007/978-3-642-40994-3_25. URL: http://dx.doi.org/10.1007/978-3-642-40994-3_25.

[6] A. Buslaev et al. "Albumentations: fast and flexible image augmentations". In: *ArXiv e-prints* (2018). eprint: 1809.06839. URL: https://albumentations.ai/.

[7] Liang-Chieh Chen et al. *Rethinking Atrous Convolution for Semantic Image Segmentation*. 2017. DOI: 10.48550/ARXIV.1706.05587. URL: https://arxiv.org/abs/1706.05587.

[8] Alexander D'Amour et al. *Underspecification Presents Challenges for Credibility in Modern Machine Learning*. 2020. DOI: 10.48550/ARXIV.2011.03395. URL: https://arxiv.org/abs/2011.03395.

[9] Logan Engstrom et al. *Exploring the Landscape of Spatial Robustness*. 2017. DOI: 10.48550/ARXIV.1712.02779. URL: https://arxiv.org/abs/1712.02779.

[10] Robert Geirhos et al. "Shortcut learning in deep neural networks". In: *Nature Machine Intelligence* 2.11 (Nov. 2020), 665–673. ISSN: 2522-5839. DOI: 10.1038/s42256-020-00257-z. URL: http://dx.doi.org/10.1038/s42256-020-00257-z.

[11] Tejas Gokhale et al. *Generalized but not Robust? Comparing the Effects of Data Modification Methods on Out-of-Domain Generalization and Adversarial Robustness*. 2022. DOI: 10.48550/ARXIV.2203.07653. URL: https://arxiv.org/abs/2203.07653.

[12] Dan Hendrycks and Thomas Dietterich. *Benchmarking Neural Network Robustness to Common Corruptions and Perturbations*. 2019. arXiv: 1903.12261 [cs.LG].

[13] Dan Hendrycks and Thomas Dietterich. *Benchmarking Neural Network Robustness to Common Corruptions and Perturbations*. 2019. arXiv: 1903.12261 [cs.LG].

[14] Dan Hendrycks et al. *AugMix: A Simple Data Processing Method to Improve Robustness and Uncertainty*. 2019. DOI: 10.48550/ARXIV.1912.02781. URL: https://arxiv.org/abs/1912.02781.

[15] Ayoung Honga et al. "Deep Learning Model Generalization with Ensemble in Endoscopic Images". In: *Proceedings of the 3rd International Workshop and Challenge on Computer Vision in Endoscopy (EndoCV 2021) co-located with with the 17th IEEE International Symposium on Biomedical Imaging (ISBI 2021)*. 2021, pp. 80–89. URL: http://ceur-ws.org/Vol-2886/paper8.pdf.

[16] Hossein Hosseini, Baicen Xiao, and Radha Poovendran. *Google's Cloud Vision API Is Not Robust To Noise*. 2017. arXiv: 1704.05051 [cs.CV].

[17] Andrew Ilyas et al. *Adversarial Examples Are Not Bugs, They Are Features*. 2019. DOI: 10.48550/ARXIV.1905.02175. URL: https://arxiv.org/abs/1905.02175.

[18] Debesh Jha et al. *Kvasir-SEG: A Segmented Polyp Dataset*. 2019. DOI: 10.48550/ARXIV.1911.07069. URL: https://arxiv.org/abs/1911.07069.

[19] Jacob Kauffmann et al. *The Clever Hans Effect in Anomaly Detection*. 2020. arXiv: 2006.10609 [cs.LG].

[20] Tsung-Yi Lin et al. "Feature Pyramid Networks for Object Detection". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. July 2017.

[21] Alexander Robey, Hamed Hassani, and George J. Pappas. *Model-Based Robust Deep Learning: Generalizing to Natural, Out-of-Distribution Data*. 2020. arXiv: 2005.10247 [cs.LG].

[22] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. DOI: 10.48550/ARXIV.1505.04597. URL: https://arxiv.org/abs/1505.04597.

[23] Veit Sandfort et al. "Data augmentation using generative adversarial networks (CycleGAN) to improve generalizability in CT segmentation tasks". In: *Scientific Reports* 9 (Nov. 2019). DOI: 10.1038/s41598-019-52737-x.

[24] Bernhard Schölkopf. *Causality for Machine Learning*. 2019. DOI: 10.48550/ARXIV.1911.10500. URL: https://arxiv.org/abs/1911.10500.

[25] Juan Silva et al. "Toward embedded detection of polyps in wce images for early diagnosis of colorectal cancer". In: *International journal of computer assisted radiology and surgery* 9.2 (2014), pp. 283–293. DOI: https://doi.org/10.1007/s11548-013-0926-3.

[26] Vajira Thambawita et al. "DivergentNets: Medical Image Segmentation by Network Ensemble". In: *Proceedings of the 3rd International Workshop and Challenge on Computer Vision in Endoscopy (EndoCV 2021) co-located with with the 17th IEEE International Symposium on Biomedical Imaging (ISBI 2021)*. 2021, pp. 27–38. URL: https://arxiv.org/abs/2107.00283.

[27] Pavel Yakubovskiy. *Segmentation Models Pytorch*. https://github.com/qubvel/segmentation_models.pytorch. 2020.