Improving performance in cross-modality single-source domain generalization for medical segmentation tasks using a GRL-based pipeline with a VFFC module on a SEGRESNET

an informal exploratory write up

by Dillan Imans

02/03/2025

Table of Contents

Preface

Few months ago I joined a lab in my university that was doing medical ai. Prior to this I was already doing medical ai and so when I joined lab I just went straight into a project. Initially I think my Prof here just wanted to see if I could actually do this whole medical ai thing before he assigned me to like a proper project, so I was just given a “broad” goal of domain generalization in the medical domain to do. Few months later (that’s today), Prof told me that I should stop doing this independent project that I was doing (which is this project here) and told me that I should be doing a proper project that followed the goals of the research team in the lab. This was like few days ago as I’m writing this preface, and this new project hasn’t been given to me yet ‘cause it’s chinese new years here so, I’m kinda doing nothing right now.

Whilst I’m waiting for the new project to be given, I thought to myself, “ah it would be a waste if I just threw months of work down the barrel. how could i make use of this project”. And so from there, I thought of this. Rather than throwing all my work away, I’ll just post this up to my github with an “informal exploratory write-up” documenting all my work since the beginning. That way not only have I learned from this project, I’ve also put something up on my github so that people can see that what I’ve been doing.

As you’ve noticed already this whole thing will be written in this manner, that is informally and with bad grammar everywhere. I won’t be doing rewrites or drafts. I’ll just write the whole thing once and have it done that way. I’m not trying to write a manuscript or a conference paper, I’m just trying to have fun whilst documenting my work and showing the work I’ve done. This is with the hopes of a recruiter seeing this or some guy wanting to do exactly the same idea that I’ve done but realizing that it doesn’t work (as you will soon see :p). Alas, I still want this to be as accurate as possible, so I will try to make 0 mistakes in terms of related works, metrics, discussions, and such.

Thought Process and Hypothesis

Problem Statement

Right so what's the problem here? When we train our AI models on some dataset, most of the datasets we have (especially in the medical domain) is stuck on a single domain or a single modality. Take for example, we want to train an AI model that can segment brain tumors. If this works at full capacity, this will undoubtedly change the future of medicine. An AI model can just segment out a tumor, and a robotic surgical arm can automatically take the tumor out. Imagine the reduction in healthcare costs and the possibilities it creates for people that need medical care but does not have the ability to pay for such.

Now the cool thing is, we already have AI models that can do this. You can find many papers online that does this, having their code posted up to github, making it open source for everyone to literally use for free. Take for example this 2022 paper [1] that already has a 90 dice score on the whole brain tumor, which is already a pretty good metric for segmentation. Though the problem we have here is not so much the AI model's performance on the dataset its trained on, but moreso the AI model's performance when we test it on another domain or another modality.

Many papers show very poor performances. The people in my lab, they were able to get super high accuracies on classification on fundus images when evaluated on the SAME domain (like 99% or whatever). But when they evaluate it on another domain (fundus images that were scanned using different machines that the AI model was trained on), it got absolutely poor performances. When I say different domain, it could be different machines or different modalities. The problem that I'm focusing on is more on a different modality (say, MRI scans of t1, t2, etc), but the whole experiment and methodology SUPPOSEDLY would make sense when tried with different machines instead.

So then you could say, why don't we just train AI models with a dataset that has all the modalities and all the machines? And you're right, that would work, except the datasets that we have (especially medical datasets) are very scarse. It's not like language models where there is literally texts everywhere in the internet, medical data is so much more difficult to come across due to all the regulations with sharing and such. In this case, we researchers are left with the problem of trying to generalize our AI models to make them able to work when evaluated on different machines. Even if we train on a single domain or a single machine, we want it to have as high of a performance as possible when it is evaluated on another machine. We want this to work to achieve the end goal of being able to use our AI models across as many hospitals and as many countries as possible, thereby making this whole AI thing in medicine work properly.

Exploring people's ideas

So with the problem in mind, how do we achieve this? There has been so many papers in the internet that has published methodologies to try to achieve this. Ofcourse different papers use different datasets with different modalities and machines and goals. Unfortunately I am too lazy to compile all the papers I have read into this document. Fortunately it's super easy to find papers. What I like to do is to go on all the MICCAI (2022, 2023, 2024) papers list and literally just ctrl + f or search for keywords like "domain generalization" or "domain adaptation". You can also search for these things on other conferences like CVPR or google scholar but you'll find less "medical related" papers. Note: Domain generalization is training on one or more domains and testing on fully unseen test domains. Domain adaptation is training on one or more domains with UNLABLLED TEST DOMAINS (but seen), and then testing on the domain of the unlabelled test domain with actual labels. Our focus is domain generalization.

These papers do a variety of techniques. A lot of them do synthetic data augmentation, where they augment the original data to look as similar as possible to the test domain in order to trick the model to learn on "different domains", learning "domain invariant features" which is not intensity or colour but moreso shapes and structures. Aside from these augmentations, some do model based changes like making a model more complex or more robust in many various ways like including a module or changing certain architectural changes. Ofcourse then there are so many different ways to do this.

A thought I had before doing this vffc thing was using a diffusion model to generate synthetic data, and then training a segmentation model on this synthetic data. Though after looking deeper into this, it probably wouldn't work + the resources you need to trian a diffusion model was absolutely insane. Sure this probably works on like simpler domains like dogs cats classification or whatever, but in generating synthetic MRI scans that mirror a specific machine's intensity level, diffusion models are just not accurate enough.

So then after reading a buck load of papers, forming theories, and creating neural pathways in my monkey brain, I thought of something.

Exploring my idea - Frequency

Frequency. The main idea here is that we convert something from the spatial domain (that's just the domain we see right now like the images on 2d and 3d) into the frequency domain with a fast frequency transform which produces the frequency domain representation of our data. Then after conversion we filter the LOW FREQUENCY stuff which represents moreso the intensity and colour and keep the HIGH FREQUENCY stuff which represents moreso the structures and shapes (as it is jumps of colour instead of steady colour). After filtering we then just convert it back to the spatial domain. So the whole main idea here is that the hopes of being able to filter out the low frequency stuff which correlates to more domain variant features and keep the high frequency stuff which correlates to shapes and

structures which are more domain invariant features, which then hopefully, at the end, improves domain generalization. VOILA.

Now it's not as simple as it sounds. I actually literally tried just training a model where the low frequency components were filtered out by a constant and it's safe to say this was stupid. Yes the shapes on the 3d image was much more focused on the structures and shapes of the actual brain, but the tumor itself is like cleaned out (tumors they aren't just shapes in the brain, they have a 'smooth intensity flow' around them, so when we just delete the low frequency components, the tumor's colour get distorted and training becomes utterly useless). So instead of only filtering the low frequency components, we would need some kind of convolution mask that can mask out the components that aren't required and keep the frequency components that are actually needed.

So then that's the idea now right. We convert the thing from the spatial domain to the frequency domain so that we can represent the image frequency wise. We then pass it through a convolutional mask to filter out frequency components that aren't useful and only keep frequency components that are useful. Instead of using a constant to filter this out, we would instead train a model to do this so that the model would be able to figure out by itself what to keep and what to leave in the frequency domain. However this leaves a new issue, how can the model know what the domain invariant features are? If we just put this frequency module into a model, it would literally just learn to create masks that improve segmentation on the current domain rather than learn the "domain invariant features". So how do we solve this?

Oh yeah, about the papers. I could organize a list of papers and talk about each of them and analyze them here but honestly I am too lazy for that. Instead I'll just give you an unorganized list of papers that use frequency to help improve metrics. Through reading these papers I got my idea on using frequency to better domain generalization performances. Thank goodness I archived them.

- HF-ResDiff: High-Frequency-Guided Residual Diffusion for Multi-dose PET Reconstruction [8]
- High-Frequency Space Diffusion Model for Accelerated MRI [9]
- Single-source Domain Generalization in Deep Learning Segmentation via Lipschitz Regularization [10]
- Boosting Diffusion Models with Moving Average Sampling in Frequency Domain [11]
- Zero-shot Medical Image Translation via Frequency-Guided Diffusion Models [12]

Exploring my idea - GRL and Bezier Intensity Augmentations

So how do we force the model to learn domain invariant features? We have that frequency filter convolutional mask that masks out frequency components that are unimportant, but right now the model is making the mask to better the performance for segmentation as it is directed by a segmentation loss. How do we force the model to learn to segment, but doing it in such a way that it segments well in other domains as well? (a.k.a

learning to segment tumors through domain invariant features rather than domain variant ones).

The idea is to use a gradient reversal layer with a domain classifier. If you want an indepth discussion on this, check this out [2]. But essentially how this works is that we put another loss aside from our main segmentation loss. This loss will be made by a domain classifier model that takes the decoded features (after it has been passed through the encoder) that tries to classify the image's domain (which means that in training, we will have more than 1 domain, can be synthetic). Though if we pass a normal loss (minimizing loss) then the domain classifier will learn well to differentiate domains, and that would not help the feature extractor model (the segmenter) to learn domain invariant features. What we need to do is to give the main segmenter model the NEGATIVE LOSS of the domain loss. So then from this, the domain classifier will be confused when it gets features from the feature extractor on what the domain is. This then makes the feature extractor be forced to LEARN DOMAIN INVARIANT FEATURES, due to the negative loss that it's given. With this in mind, the feature extractor model (with the frequency module) will then be able to learn masks that filter out unimportant features that relate to the domain and keep DOMAIN INVARIANT FEATURES. Does this make sense? It made sense to me, but when I did my experiments, the results are….well….let's just see later.

So then if we use this method, we would need more than one domain right? Right. But we want this to be single source domain generalization, not double or triple source. So one way to use only a single source but be able to have more than 1 domain is to create synthetic domains (remember the thing I talked about above?).

Now we want these domains to actually represent something. If we only do augmentations like flips and rotates, they mean nothing. We have these augmentations done in anyways 'cause it just helps to have flips and rotates for not overfitting the model. We want the domains to be synthetic, to be made from the single source training domain that we have but augmented in some way. What augmentations other than flips and rotates? Intensity.

Check out the figures below. They are all 4 modalities that we are going to work with. The goal is to train on one of the modalities (take for example t2) and evaluate on other domains (take for example t1, t1ce, and flair). The biggest difference from one modality to the others is their colour, their intensity. So it would make sense for us to create synthetic domains through intensity augmentations right?

Yes. Infact [3] which uses a similar dataset uses bezier curve nonlinear intensity transformations to create synthetic domains, found in [4], which shows that this is possible. Basically since all modalities have differences specifically on modality, we want our synthetic domains to be intensity differences. So we use this bezier curve thing to make different domains, and force the model to learn domain invariant features from these different synthetic domains. Voila.

Hypothesis

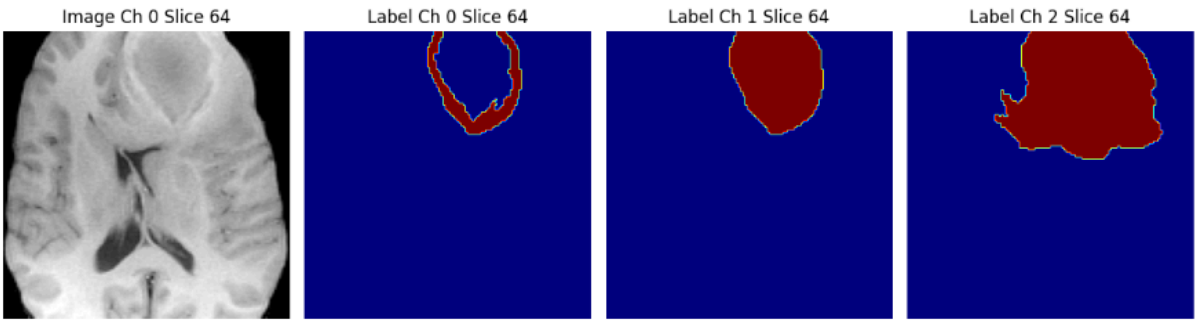
By integrating a frequency-based filtering module with a Gradient Reversal Layer (GRL) and domain classifier, and augmenting the training data using Bezier curve-based intensity transformations, we can improve the domain generalization capabilities of AI models in medical image segmentation tasks. Specifically, the model will learn to focus on domain-invariant features (e.g., shapes and structures) while suppressing domain-variant features (e.g., intensity and color), thereby achieving better performance when evaluated on unseen domains or modalities.

Methodology

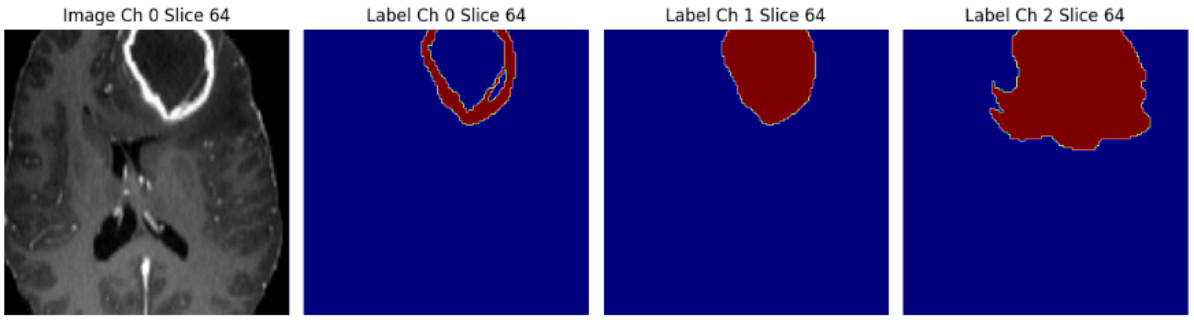
So now I'll talk specifically how we do this and what we do it with.

Dataset Introduction

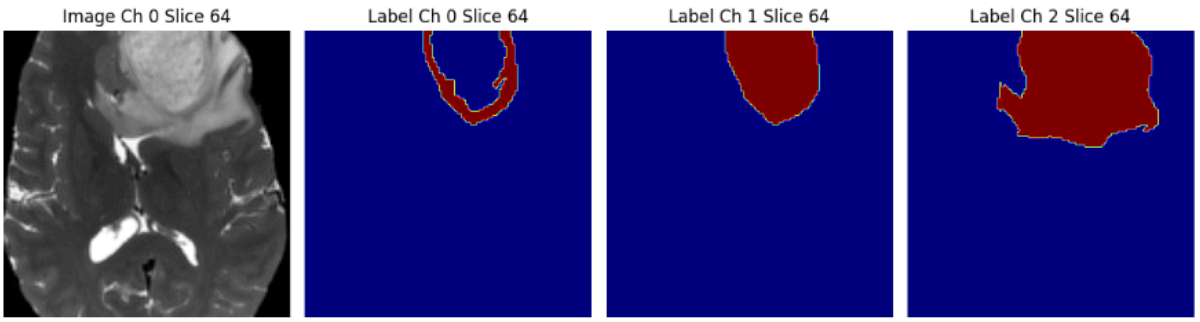
Dataset used is the BraTS2021 dataset which is a 4 MRI modalities, brain tumor segmentation dataset. There are about 1000ish patients in this dataset, with each patient having 4 different MRI scans from different MRI modalities. These modalities are t1, t1ce, t2, and flair.



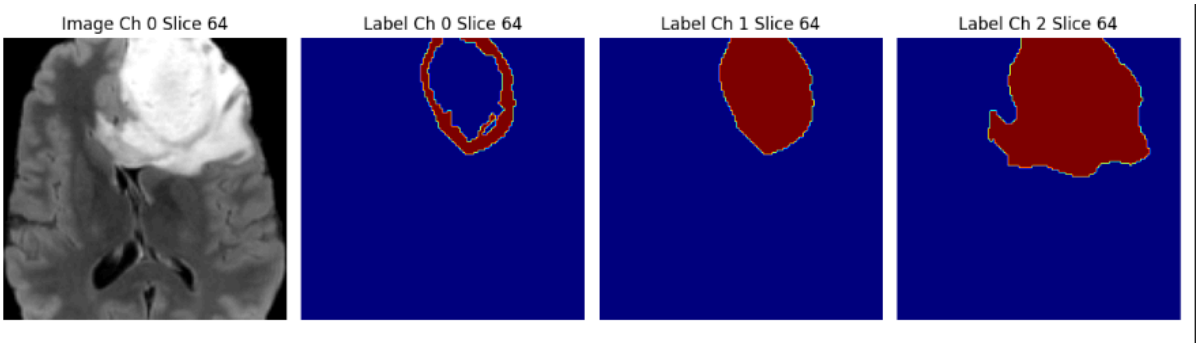
t1



t1ce



t2



flair

The above photos are taken from the same patient. They show the differences in each modality, and as we can see they are mostly intensity and colour differences. We want the model to be forced to learn not from the colour and intensity, but from the shapes of the tumor instead. Now, the 3 labels on the right are the different tumour labels: enhancing tumor, tumor core, and whole tumor. When training and evaluating this model, we are going to be training it on 3 different labels with each of their own dice scores. To specify further, the exact spatial dimensions of each patient for all 4 modalities is the exact same. You may see some differences above because of my random padding and random cropping, which we'll talk about now.

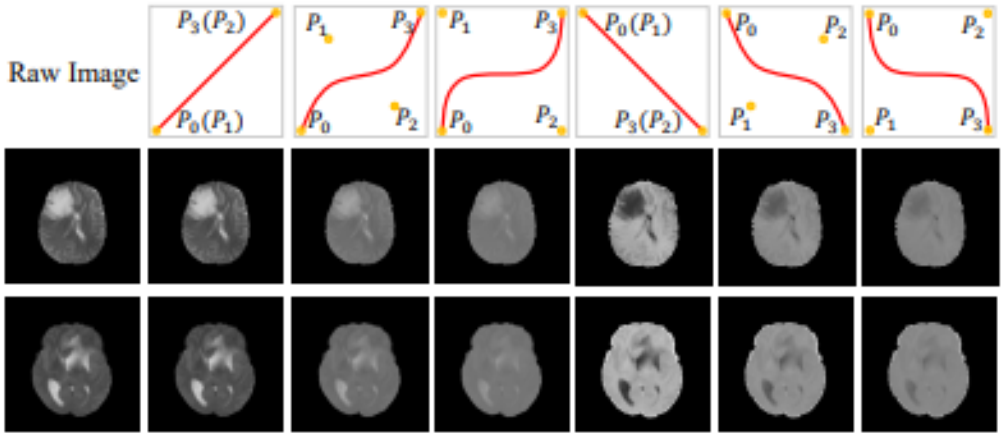
Preprocessing the Dataset

First we need to take only ONE modality. Let's take t2. From this t2 modality, we split into subsets of domain 0 domain 1 and domain 2. Each subset has a collection of different patients, e.g 0-300 for domain 0, 301-601 for domain 1, 602-902 for domain 3. Different patients for different subsets.

We will need to do some specific preprocessing for the individual data here. Initially the size of each data is 240x155x155, which is huge. To be able to fit into gpus, we'll need to crop it. We do this by cropping to non-zero regions which is to crop the image and label into the smallest bounding box that contains all non-zero voxels. We can then normalize. We want to make sure all the data have the same size, so crop/pad into 128x128x128.

The augmentation will be random flipping and random rotation for all domains. This is just to help overfitting. Nothing special.

Then we do the bezier curve transformations. So to simulate domain variations across the three domains, we will need to transform each domain in their own ways. We will have domain 0 as the default domain with no bezier curve transformations. Domain 1 will then take control points `[-1, -1], [-v, v], [v, -v], [1, 1]` which is in the positive gradient for the bezier curve whilst domain 2 will take `[1, 1], [w, -w], [-w, w], [-1, -1]` which is in the negative gradient. Both v and w are random values from -1 to 1 to create variation. Now to understand deeply this concept, you'll need to go deep into the maths in this paper [3], but I can show you a quick figure here to let you see what happens in these transformations.



This is taken from [3]. Basically the 2nd to 4th are positive gradient transformations which is our domain 1. The 5th to 7th are negative gradients which is our domain 2. You can see that they are transformed similarly, but with some variations. We want to do this for our single source to imitate the intensity differences for our modalities.

Now, these preprocessing steps are flexible. You can instead have domain 1 and domain 2 and leave out the "base domain" to make it 50:50 for the subset. You can disable the random flipping and rotation augmentations. You can adjust the curves. Infact whilst I'm writing this I'm experimenting with different augmentations to see the best performance. The one above is just the "most complete" one so you can see the options that I had when I did my experiments.

Base model and Frequency Module

Initially I tried using the 3DUNet because of the simplicity of it. Though when I did experiments with it, it had a really poor generalization performance. I then tried the SegResNet[5] without the VAE, which I believe was SOTA by the time I'm writing this. And yes it did so much better.

The frequency module was based on the volumetric fast fourier convolution (VFFC) [6] which was the 3D version of [7]. This module basically passes the spatial domain data through a fast fourier transform to transform it into the frequency domain, push it through some resblocks, then pass it back through an inverse fast fourier transform to transform it back into the spatial domain. This is with the hopes that resblocks in the frequency domain combined with the GRL will force the model to learn to mask out low frequency components. We put this module into the bottleneck of the model (the deepest part where it is at the highest representation level).

One thing to note on the VFFC is the local vs global branch. Basically, when we pass features into the VFFC, it'll go through two paths: the local branch which cuts the features to patches vs the global branch which operates on the whole thing. This ratio of how much goes to the local branch and how much goes to the global branch is editable. It is considered as a hyperparameter that we need to optimize by ourselves.

Experimental Setup

I'll keep this short.

- Learning rate: 1e-4, with CosineAnnealingLR
- Weight decay: 1e-5
- Epochs: 100
- Lambda (controlling how much of the domain loss is being fed): With a scheduler, goes to max on max epoch.

Uses torch.distributed for multi-GPU setup. Used 4x 2080s for this.

Results and Evaluations

Evaluated on Dice Scores.
Written as (Mean) - (Whole Tumor) | (Enhancing Tumor) | (Tumor Core)

3DUNET

Dice Scores	T1 test	T1CE test	T2 test	Flair Test	Highest Own Val
4 modalities stacked	x	x	x	x	0.73 - 0.73 | 0.69 | 0.75
T2 Trained (no aug)	0.11 - 0.08 | 0.09 | 0.15	0.08 - 0.04 | 0.09 | 0.11	0.62 - 0.75 | 0.49 | 0.62	0.11 - 0.19 | 0.04 | 0.10	0.56 - 0.56 | 0.44 | 0.69

SEGRESNET

Dice Scores	T1 test	T1CE test	T2 test	Flair Test	Highest Own Val
T2 Trained (no aug)	0.04 - 0.03 | 0.05 | 0.04	0.29 - 0.28 | 0.39 | 0.19	**0.74 - 0.57 | 0.74 | 0.89**	**0.65 - 0.47 | 0.67 | 0.83**	**0.68 - 0.68 | 0.53 | 0.82**
T2 Trained (bezier intensity)	**0.05 - 0.04 | 0.06 | 0.05**	**0.30 - 0.28 | 0.39 | 0.22**	0.70 - 0.53 | 0.71 | 0.87	0.63 - 0.42 | 0.64 | 0.83	0.63 - 0.63 | 0.48 | 0.80
T2 Trained (vffc 3 dom)	0.03 - 0.02 | 0.03 | 0.05	0.10 - 0.09 | 0.15 | 0.07	0.71 - 0.55 | 0.72 | 0.88	0.58 - 0.39 | 0.59 | 0.78	0.66 - 0.66 | 0.49 | 0.82

Special case - SEGRESNET

Dice Scores	T1 test	T1CE test	T2 test	Flair Test	Highest Own Val
T1CE T2 (bezier intensity)	0.06 - 0.05 | 0.07 | 0.07	0.59 - 0.69 | 0.72 | 0.35	0.44 - 0.20 | 0.39 | 0.74	0.54 - 0.33 | 0.48 | 0.80	0.82 - 0.85 | 0.79 | 0.81

We see first the 3DUNET. Immediately we can see that the model's ability to generalize to other modalities is absolutely horrendous. While the T2 test and val is decent, the ability for the model to evaluate on other modalities is absolute garbage. Compare it to the same pipeline used with the SEGRESNET model, we can see drastic improvements in dice scores.

Going to the SEGRESNET results, we can finally see why my methods don't work :(. Basically the no aug pipeline does better when evaluating against the domain itself (this makes sense as the model would fit to this domain) and to flair. The bezier intensity pipeline (which is just random bezier transformations with no GRL and no VFFC) does slightly better in t1 and t1ce, but not enough to justify any relevant improvements. The main method, which uses GRL and VFFC does worst for all domains. The whole goal is to improve the metrics on all other domains (t1, t1ce, and flair) but instead we lost performance, especially in t1ce. This just shows that my methodology does not work and why this whole idea went into an "informal explanatory write-up" instead of a MICCAI conference paper. It's alright though I learned alot from this.

As an additional evaluation, I tried training a special case which is the T1CE and T2 domains together. We still see poor performances in T1 and FLAIR here, which is a surprise to me because I thought adding more domains here would improve atleast the T1 or FLAIR dice scores. So now I'm like, huh. But anyways this isn't really my concern. My concern is why my VFFC doesn't work huhuhu sosad.

The Opposite of Preface

There we are. I highly doubt you read through all that as it was mostly just me blabbering about stuff that in the end DOES NOT WORK. Though even so I'm happy for what I've made. The main goal was to reach MICCAI 2025, but as of writing this I only have 3 days left to submit an abstract, and my experiments did not succeed.

Aside from all the topics, theories, coding, and experiments that I've learned throughout the journey, I learned that in research, it's okay to fail sometimes. I've spent around 3 months figuring out and building this, and at the end I failed, and that's okay. As of writing this, I'm still in the 3rd year of my undergraduate, and I'm still trying to learn as much stuff as possible in this field (though I'm eyeing more on bioinformatics right now cuz I wanna go masters in bioinformatics).

If you did read all of this, or even only some of it, I'd like to say thanks. I hope you atleast get something out of it.

References

Github link: https://github.com/DillanImans/brats_grl_vffc

[1] Peiris, H., Chen, Z., Egan, G., & Harandi, M. (2021, September). Reciprocal adversarial learning for brain tumor segmentation: a solution to BraTS challenge 2021 segmentation task. In *International MICCAI Brainlesion Workshop* (pp. 171-181). Cham: Springer International Publishing.

[2] Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, H., Laviolette, F., ... & Lempitsky, V. (2016). Domain-adversarial training of neural networks. Journal of machine learning research, 17(59), 1-35.

[3] Zhou, Z., Qi, L., Yang, X., Ni, D., & Shi, Y. (2022). Generalizable cross-modality medical image segmentation via style augmentation and dual normalization. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 20856-20865).

[4] Zhou, Z., Sodha, V., Rahman Siddiquee, M. M., Feng, R., Tajbakhsh, N., Gotway, M. B., & Liang, J. (2019). Models genesis: Generic autodidactic models for 3d medical image analysis. In Medical Image Computing and Computer Assisted Intervention–MICCAI 2019: 22nd International Conference, Shenzhen, China, October 13–17, 2019, Proceedings, Part IV 22 (pp. 384-393). Springer International Publishing.

[5] Myronenko, A. (2019). 3D MRI brain tumor segmentation using autoencoder regularization. In Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries: 4th International Workshop, BrainLes 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 16, 2018, Revised Selected Papers, Part II 4 (pp. 311-320). Springer International Publishing.

[6] Quattrini, F., Pippi, V., Cascianelli, S., & Cucchiara, R. (2023). Volumetric fast fourier convolution for detecting ink on the carbonized herculaneum papyri. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 1726-1734).

[7] Chi, L., Jiang, B., & Mu, Y. (2020). Fast fourier convolution. Advances in Neural Information Processing Systems, 33, 4479-4488.

[8] Tang, Z., Jiang, C., Cui, Z., & Shen, D. (2024, October). HF-ResDiff: High-Frequency-Guided Residual Diffusion for Multi-dose PET Reconstruction. In International Conference on Medical Image Computing and Computer-Assisted Intervention (pp. 372-381). Cham: Springer Nature Switzerland.

[9] Cao, C., Cui, Z. X., Wang, Y., Liu, S., Chen, T., Zheng, H., ... & Zhu, Y. (2024). High-frequency space diffusion model for accelerated mri. IEEE Transactions on Medical Imaging.

[10] Arslan, M. F., Guo, W., & Li, S. (2024, October). Single-source Domain Generalization in Deep Learning Segmentation via Lipschitz Regularization. In International Conference on Medical Image Computing and Computer-Assisted Intervention (pp. 666-674). Cham: Springer Nature Switzerland.

[11] Qian, Y., Cai, Q., Pan, Y., Li, Y., Yao, T., Sun, Q., & Mei, T. (2024). Boosting Diffusion Models with Moving Average Sampling in Frequency Domain. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 8911-8920).

[12] Li, Y., Shao, H. C., Liang, X., Chen, L., Li, R., Jiang, S., ... & Zhang, Y. (2023). Zero-shot medical image translation via frequency-guided diffusion models. IEEE transactions on medical imaging.