DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics

Image retrieval based brain tumor modeling

Marcel Thomas Rosier

DEPARTMENT OF INFORMATICS

TECHNISCHE UNIVERSITÄT MÜNCHEN

Bachelor's Thesis in Informatics

Image retrieval based brain tumor modeling

Image-retrieval basierte Gehirntumormodellierung

Author:	Marcel Thomas Rosier
Supervisor:	PD Dr. rer. nat. Tobias Lasser
Advisor:	M.Sc. Ivan Ezhov
Submission Date:	15.02.2022

I confirm that this bachelor's thesis in informatics is my own work and I have documented all sources and material used.

Munich, 15.02.2022 Marcel Thomas Rosier

Contents

1 Introduction **1**

2 Related Work **2**

3 Approach **3**
3.1 Ground Truth Dataset . . . 3
3.2 Learn-Morph-Infer Pipeline . . . 4
3.3 Baseline . . . 5
3.4 Compression . . . 6
3.4.1 Down Sampling . . . 6
3.4.2 Autoencoder . . . 7
3.4.3 Variational Autoencoder . . . 10

4 Evaluation **14**
4.1 Quantitative Evaluation . . . 14
4.1.1 Baseline . . . 14
4.1.2 Compression . . . 15
4.2 Qualitative Evaluation . . . 21
4.2.1 General . . . 21
4.2.2 Autoencoder & Variational Autoencoder . . . 23
4.3 Performance in the Pipeline Context . . . 26

5 Future Work **28**

6 Conclusion **29**

Bibliography **30**

1 Introduction

Magnetic resonance imaging (MRI) is the state of the art diagnostic tool to diagnose and locate brain tumors. While being able to reliably highlight areas of sufficiently high tumor cell concentration, it struggles with areas of lower density. For glioblastoma (GBM), which is one of the most aggressive types of brain tumor that characteristically infiltrates surrounding tissue, the shortcomings of MRI scans often lead to tumor recurrence after treatment.
Current treatment plans try to account for the unknown infiltration by also uniformly targeting the visible tumor's surrounding. While decreasing the probability of secondary tumors, this has the significant drawback of typically still not resecting all infiltrations and often unnecessarily damaging healthy tissue, which has a negative impact on the patient's life quality. Personalizing the target of radiotherapy by complementing the MRI scans with individual simulations that give information about the spatial distribution of tumor cell concentration could preserve healthy tissue and reduce the likelihood of residual tumor cells [Stu+14] [Lip+19] [AC17].
Conventional approaches to implement a personalization try to simulate the tumor growth for each individual patient based on differential equations. However, utilizing highly efficient numerical solvers still results in extreme runtimes that obstruct transfer into clinical practice [Ezh+21a]. To overcome this issue, [Ezh+21b] introduced the learn-morph-infer pipeline that delivers fairly accurate results in near real-time. Since it relies on a neural network to predict patient-specific parameters, it might however not be robust to all inputs.
In the following, we propose an image retrieval based approach that performs a query of patient-specific scans to a database of synthetic tumors, returning the closest resemblance of the patient's tumor. This procedure can function as a replacement for the predicting network and forward solver within the pipeline. As a baseline, this image retrieval process is implemented via a primitive iterative pairwise comparison. We investigate how down sampling, as well as compression using autoencoders (AE) and variational autoencoders (VAE), can improve runtimes while assessing the quality of results by comparison with the iterative baseline. Moreover, we point to further possible compression methods as well as overall improvement options.
As a result, our work shows that - given a sufficiently advanced base dataset - our query approach can yield useful and, depending on the chosen optimization, also deterministic results in the order of seconds. Moreover, we establish that deep learning based encodings can drastically compress data while preserving similarity relations up to a certain degree.

2 Related Work

Our proposed method is a modification of the existing learn-morph-infer pipeline [Ezh+21b] and therefore closely related to it and corresponding papers [Ste20] [Sci21].
The basis for our work is the synthetic dataset generated by [Ste20], which utilizes software and results from [LE20] [Lip+19].
Our final results are presented similarly as in [Ezh+21b].
The author has studied about autoencoders mainly from [GBC16] and about variational autoencoders from [Hig+] and [KW13].
Losses have been plotted using TensorBoard, the seaborn library and MatPlotLib. All tumor visualizations have been created with either MatPlotLib or the Mayavi framework [RV11].

3 Approach

3.1 Ground Truth Dataset

The basis for our work is a dataset of 100,000 synthetic tumors of resolution 128^3, created by [Ste20]. In order to generate synthetic tumors, a common brain anatomy is required that is able to represent a broad range of realistic human brains. For this purpose, the atlas brain anatomy introduced by [Roh+10] was utilized, which represents a statistical average for the relevant distributions of cerebrospinal fluid (CSF), white matter (WM) and gray matter (GM), combined into a single atlas space. In this atlas space, a tumor can then be simulated using the reaction diffusion model by setting values for the initial location (x, y, z), the proliferation rate ρ, the infiltration rate in WM D_w as well as the simulation end time T_{end}, which corresponds to the tumor's age. The synthetic dataset was generated by randomly sampling patient-specific parameters from the subsequent ranges and feeding the resulting parameter sets $\theta = (x, y, z, D_w, \rho, T_{end})$ to the reaction diffusion model implemented by [Lip+19].

$$
\begin{gathered}
D_w \in [0.0002\frac{cm^2}{d}, 0.015\frac{cm^2}{d}], \\
\rho \in [0.002\frac{1}{d}, 0.2\frac{1}{d}], \\
T_{end} \in [50d, 1500d], \\
x \in [0.15, 0.7], y \in [0.2, 0.8], z \in [0.15, 0.7]
\end{gathered}
$$

To further improve realism and overall quality of the data, tumors that would likely not occur in a clinical setting due to their size have been removed based on minimal and maximal size thresholds. The used thresholds were derived from the BraTS dataset as defined in [Men15].
Due to the time and therefore also runtime limitations of our project, only a random subset of 50,000 tumors has been used for all experiments throughout this project, which will be referred to as S.
For testing the quality and practical applicability of our approach, we used a dataset of 62 MRI scans of patients diagnosed with glioblastoma provided by the MRI TUM Klinikum Rechts der Isar. Based on these scans, we generated a dataset P containing binary segmentations representing the FLAIR (P^{FLAIR}) and T1Gd (P^{T1Gd}) modalities.

3.2 Learn-Morph-Infer Pipeline

The main goal of this paper is to propose an alternative step within the learn-morph-infer pipeline introduced by [Ezh+21b] that ideally leads to more deterministic and reliable results. Thus, it is indispensable to first understand the original pipeline, which consists of four main steps:
First, a patient MRI scan of a tumor observation $Y = \{y^{T1Gd}, y^{FLAIR}\}$ is registered to the atlas brain anatomy [Roh+10], yielding a transformation matrix M. Afterwards, M is used to morph Y into the atlas space, resulting in Y_{atlas}. Subsequently, a neural network, that was pretrained on the dataset seen in section 3.1 to learn the function $Y_{atlas} \mapsto \theta_c$, infers a set θ_c of patient-specific parameters, which is then used as an input for a tumor solver in step 3 to generate a patient-specific tumor simulation in the atlas space. As a last step, utilizing the inverse of M, the atlas simulation is morphed back to the patient brain anatomy to obtain the final patient-specific simulation [Ezh+21b].

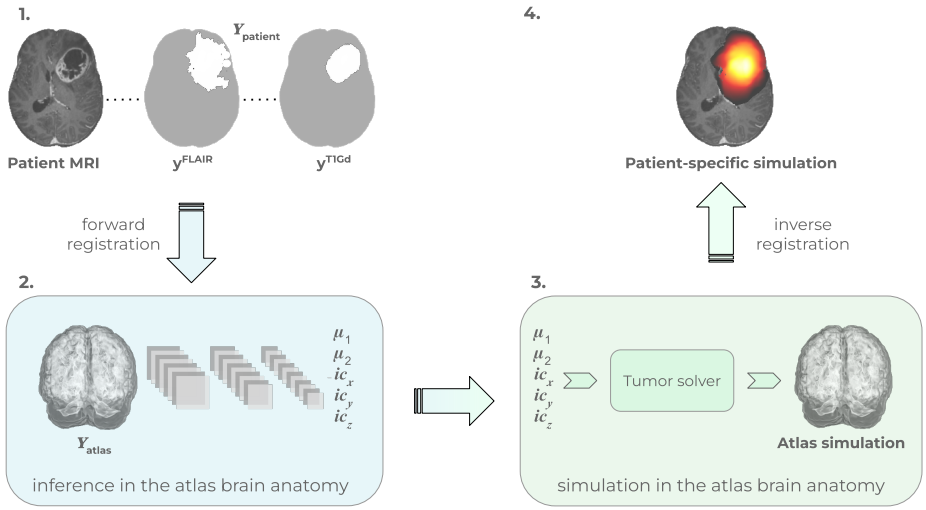

Figure 1: A sketch showcasing the learn-morph-infer pipeline. A patient MRI scan is registered to the atlas space (1). Then, a neural network infers patient-specific parameters from the morphed scans (2), which serve as input for a tumor solver that yields a tumor simulation in the atlas space (3). Finally, the output is morphed back to the patient brain anatomy to obtain the patient-specific simulation (4).

Implementations of this pipeline (e.g. [Sci21]) are able to produce fairly accurate results (average dice scores of roughly 0.9) in the order of seconds. However, neural networks always come with the drawback of being unpredictable for unseen data, which introduces a potential risk.

In the following, we propose a deterministic approach to overcome this limitation. Instead of using a neural network to infer θ_c, which can then be passed to a tumor solver, we suggest making a simple query to a database of synthetic tumors and returning the closest

match as a result. The rest of the pipeline, namely the forward and reverse registration, remains unchanged. The following sections will discuss different ways to implement and improve this surrogate.

3.3 Baseline

The most basic way to implement the change to the learn-morph-infer pipeline proposed in section 3.2 and visualized in Figure 2 is by naively looping over the dataset and performing a pairwise comparison between input and database tumors. Importantly, the input MRI scans consist of 2 segmentations, while the synthetic tumors are spatial distributions containing continuous values in the range from 0 to 1. Therefore, we have to extract the desired segmentations from the raw synthetic tumor data. For this purpose, we threshold the synthetic tumors at the level 0.2 to receive the FLAIR and at 0.6 for the T1Gd segmentations. The used thresholds are based on common values in literature [Lê+16]. After preprocessing the dataset to make it comparable, we need to select a fitting metric. The dice coefficient (dice) is a common option for medical volume segmentations [TH15] and was therefore utilized as the main metric. It produces scores in the range $[0; 1]$, with 1 being the optimal. In order to evaluate similarities between the compressed vectors in the following sections, we additionally use the L2 $= \|\cdot\|_2$ metric (euclidean distance). Since we have to compare two segmentations for each tumor, it is possible that the separate queries result in different best matches. Hence, the 2 query results have to be combined to determine a joint best match. For simplicity, we decided to plainly add the 2 individual similarity values and choose the highest combined value for dice (highest overlap) and respectively the lowest for L2 (lowest distance) as the best match. We implemented this basic brute-force loop in a python script to create a ground truth that serves as a reference for later improvements and experiments. Since the individual comparisons are not dependent on each other, we made a first optimization by parallelizing the loop to shorten its runtime.

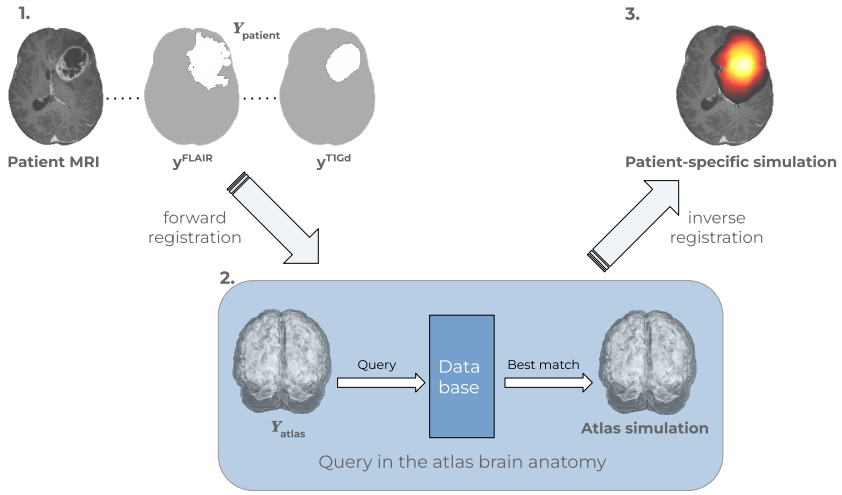

Figure 2: A sketch showcasing the proposed adaption to the learn-morph-infer pipeline. A patient MRI scan is registered to the atlas space (1). Subsequently, a query to a database of tumors is executed to get the closest match for the morphed scans (2). This best match is then morphed back to patient brain anatomy to obtain the patient-specific simulation (3).

3.4 Compression

Although parallelization can help to reduce the time complexity of the database query, the speed of this operation is still limited by the significant memory usage. This restriction also prohibits the use of efficient query frameworks. For instance, the FAISS library is designed for vector sizes in the range of roughly 20 - 2000 [JDJ17]. However, a plain flattening of our 3D volume of 128^3 exceeds this suggested limit by a factor of more than 1000. Therefore, we decided to experiment with a compression of the data. As a first step, we try a naive down sampling of the volume to lower resolutions in subsection 3.4.1. To further minimize the data size, we experimented with autoencoders and variational autoencoders and examined if these techniques are capable of creating a problem-specific encoding that might be able to sufficiently preserve similarity relationships between tumors, while drastically reducing the runtime.

3.4.1 Down Sampling

A potential optimization that is still quite close to the baseline is down sampling. Due to the tumors' relatively high resolution of 128^3, a lot of data has to be loaded and compared. This overhead can be reduced by down sampling the data to lower resolutions. In our case, we tried a down sampling to 64^3 and 32^3. The down sampling process is achieved by applying a spline interpolation of order 0 to the volume, for which we utilized SciPy's *ndimage.zoom* function [Vir+20]. For simplicity, we applied this down sampling step within the query.

However, this could also be done as a preprocessing step for the entire dataset S, leading to an additional runtime and memory usage boost since in that case only the input tumor would have to be down sampled on the fly.

3.4.2 Autoencoder

An autoencoder is a deep learning method that tries to reconstruct the input data after being put through a hidden layer h, often also referred to as latent space, that represents an encoding. Therefore, it consists of an encoder function $h = enc(x)$ and a decoder $r = dec(h)$ producing a reconstruction r. The goal is to minimize the difference between x and r while restricting the hidden layer to be of a smaller dimension than the input, producing a bottleneck. Ideally, the autoencoder learns to extract meaningful features from the input data, since it is forced to prioritize input features based on their relevancy for reconstruction [GBC16].
The core idea for our use-case is to train the autoencoder to extract meaningful properties of the tumors and then apply the encoder function to the dataset S, yielding an encoded dataset S_{enc}. For each query, we utilize S_{enc} and encode the input tumor using the same encoder, resulting in a data size drop from 128^3 to the dimension of the latent space per tumor. Since the segmentations T1Gd and FLAIR differ in their properties, two separate autoencoders are needed, leading to consequently two encoded datasets S_{enc}^{T1Gd} and S_{enc}^{FLAIR}.

Neural Network

The base architecture is a convolutional neural network (CNN). In the decoder, we scale down the volume layer-by-layer using strided 3D convolutions (stride=2, kernel size=3) followed by a convolution without stride, while increasing the number of channels. After down scaling the volume, we flatten the features and apply a linear layer to obtain a latent representation. Afterwards, the decoder mirrors the encoding process with strided 3D transposed convolutions.
In order to determine a fitting value for the smallest resolution, multiple test runs have been executed, which suggested that a down scaling to a resolution of 16^3 before flattening the volume yields the most reasonable results. This minimal resolution was used throughout all consecutive trainings. The final architecture can be seen in Figure 3.

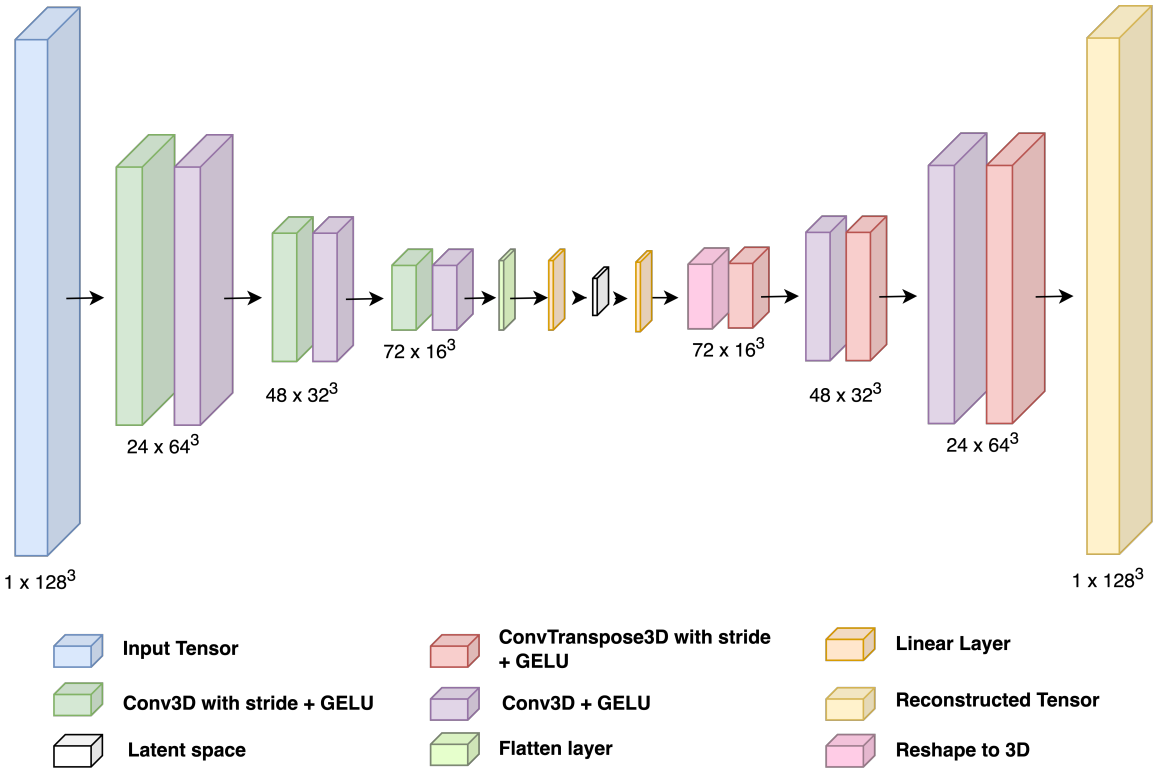

Figure 3: Autoencoder network showcasing order and type of individual layers.

Training

The network shown in Figure 3 was finally trained with both S^{T1Gd} and S^{FLAIR} as input. However, due to the similarity of the two segmentations, we used only T1Gd to tune hyperparameters in the following and afterwards transferred the results to the second modality. Since larger training set sizes showed no significant improvements, we used a relatively small training set of 1500 tumors along with a validation set of size 150. Experiments suggested a learning rate of $1e^{-5}$ combined with the Adam [KB17] optimizer. Different losses, including Binary Cross Entropy (BCE), Mean Squared Error (MSE) and a variation of dice losses, were tested out of which the default dice loss from the MONAI framework [MON20] produced the best training results. In terms of batch size, small values improved train quality, leading to a final choice of 2. Another important parameter is the size of the latent space. Motivated by the need of a significant data compression, coupled with the excessive GPU memory usage (> 48 GB) of networks with fully connected layers beyond a latent space size of 4096, we chose the latter as an upper bound. Thus, latent space sizes l in the range $l \in [1; 4096]$ were tested for 120 epochs each, producing train and validation loss performances seen in Figure 4. Although latent space sizes of 4096 and 2048 achieved better training loss results in the given comparison, we chose a size of 1024. This was motivated by the significantly smaller data mass, which justifies the slightly worse training and validation loss performance.

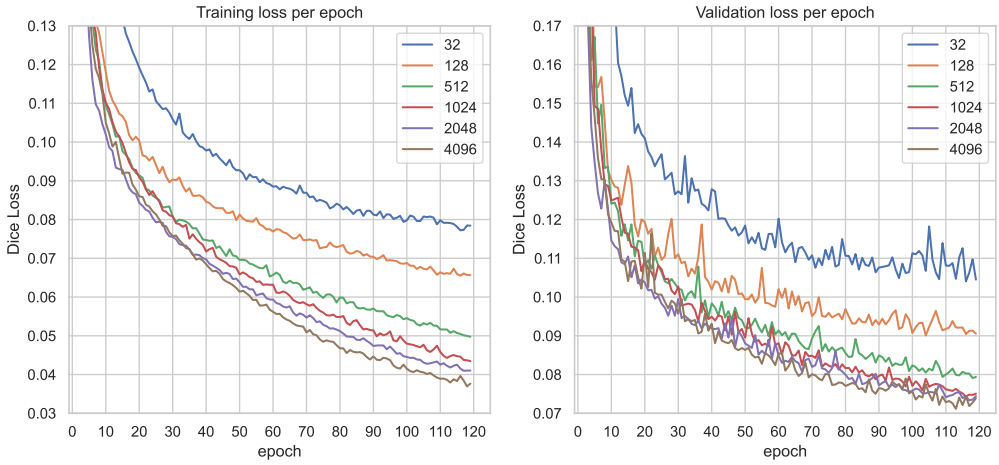

Figure 4: Comparison of average training and validation loss across all batches per epoch for a selection of latent space sizes. The validation loss is calculated at the end of each epoch, hence it can be better than the training loss in certain epochs.

Figure 5 shows the convergence of our final autoencoder networks with an increased number of epochs for both segmentations. Interestingly, both train and generalization loss perform even better, when transferred to the FLAIR segmentation. For the following evaluation in chapter 4 we chose the last epoch that provided the best validation loss (roughly 0.063 for T1Gd and 0.050 for FLAIR) and therefore generalization capabilities.
All trainings have been executed using the PyTorch framework. The larger networks with a

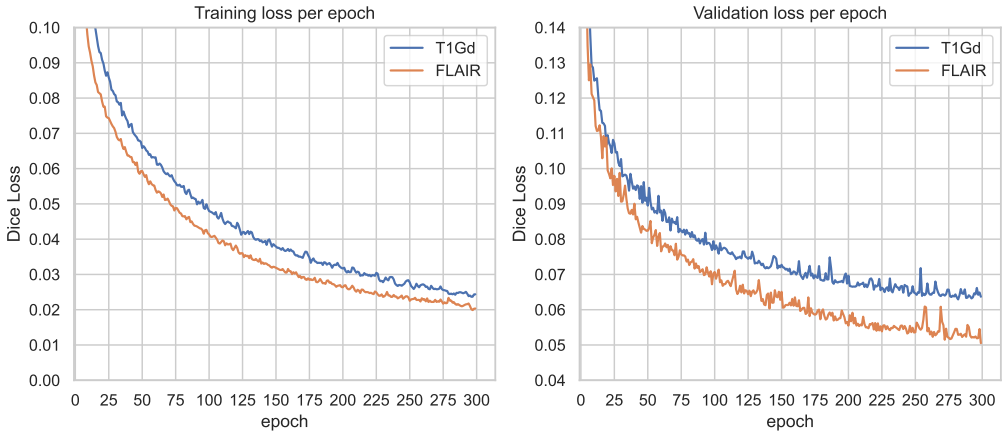

Figure 5: Comparison of average training and validation loss across all batches per epoch for the final architecture trained on the T1Gd and FLAIR segmentation.

bigger latent space were trained on an NVIDIA Quadro RTX 8000 while the smaller networks were executed on either an NVIDIA Quadro RTX 6000 or an NVIDIA Quadro P5000.

3.4.3 Variational Autoencoder

Trying to further improve the performance of our autoencoders we experimented with variational autoencoders (VAE). While being simplistic, basic autoencoders have the potential disadvantage of offering limited control over the latent space, which is the most important element for our use case. VAEs enforce a distribution over the latent space by penalizing deviations from a previously selected distribution during training, which can produce more continuous and organized latent spaces while still providing adequate reconstructions [KW13][Hig+]. Usually, this property is used to generate previously unseen output by randomly sampling from the latent space. However, we investigated if this characteristic is also beneficial for similarity preservation.

Neural Network

For comparability and simplicity, the VAE network mostly replicates the autoencoder network from section 3.4.2. The key difference is that the last linear layer before the latent space is replaced by 2 separate linear layers that receive the same input and output the mean μ and the logarithm of the variance $\log \sigma^2$. Since basic stochastic sampling is not differentiable and thereby not suitable for the back propagation during training of a neural network, the reparameterization trick [KW13] is applied. This transformation separates the stochasticity from the updating process (making back propagation possible) and outputs a latent space vector that can be used as an encoding for our application. The rest of the network remains unchanged. The adaption is visualized in Figure 6. Our implementation of the VAE specific parts is similar to the MONAI implementation [MON20], which itself is based on [KW13].

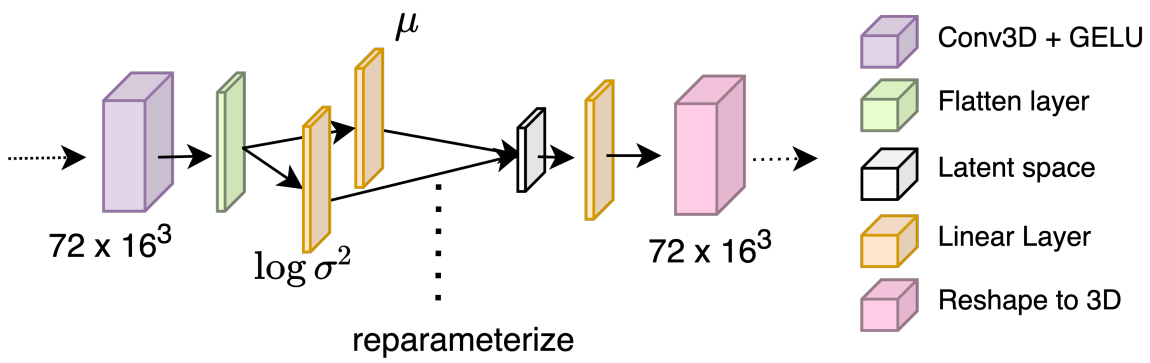

Figure 6: Adaptions to the autoencoder network from Figure 3. The last linear layer is replaced by two separate linear layers that receive the same input and output μ and $\log \sigma^2$, which are then reparameterized, resulting in the latent space vector. The rest of the architecture remains unchanged.

Training

A few aspects, such as training and validation set size, batch size and the optimizer have been reused from section 3.4.2. We also repeat the approach of first tuning the network for S^{T1Gd} and then reusing the results for S^{FLAIR}. Since a VAE also has to consider the latent space distribution during training, the loss function has been adapted to include the Kullback–Leibler divergence D_{KL} (a measure of how similar two distributions are) with a weighting factor of β [Hig+]. Therefore, the combined loss is calculated as $loss = DiceLoss + \beta * D_{KL}$. Basic VAEs use no such weighting factor, and therefore implicitly $\beta = 1$. [Hig+] shows that β can be used to balance latent channel capacity and independence constraints with reconstruction accuracy. Typically, $\beta > 1$ is chosen. However, our experiments showed that these common values prevented the network from properly reconstructing the tumor data, leading to a choice of a rather small factor of $\beta = 0.001$. Figure 7 shows the D_{KL} part of the loss that is already significant when compared to the dice loss part and thereby explains why such a small β is necessary. Note: This small β limits the potential of the VAE concept but was necessary to learn an encoding that allowed reconstruction.
Moreover, we increased the learning rate to $3e^{-5}$ and reassessed different values for the latent space size. Figure 7 compares training and validation loss, as well as the D_{KL} and dice part of the train loss. Although we chose a small β value, the VAE still significantly improved the distribution of the latent space. All latent space sizes achieved values of < 0.09, while a reference run that calculated D_{KL} without considering it in the back propagation led to D_{KL} values tending to infinity, especially for small latent spaces. In general, the majority of smaller latent sizes produce better results than larger ones in every aspect. While this was certainly expected for the D_{KL} loss, it was not obvious that the change in the network architecture would also allow the reconstruction loss to be kept minimal in small latent spaces. While we did not utilize the latent size with the best pure dice score for the autoencoder approach, we

do use the best dice score performance for the variational autoencoder with a latent space size of 8. This difference in methodology is motivated by the fact that latent spaces < 8 do not provide the significant memory advantage, which was a factor in the previous scenario.
Figure 8 shows the convergence of our final VAE network trained with an increased number of epochs for both segmentations. When transferring the T1Gd network optimizations to the FLAIR data, we can again observe an even better performance. For the following evaluations in chapter 4 we chose the last training epoch of both networks.
All VAE trainings have been executed using the PyTorch framework on either an NVIDIA Quadro RTX 6000 or an NVIDIA Quadro P5000.

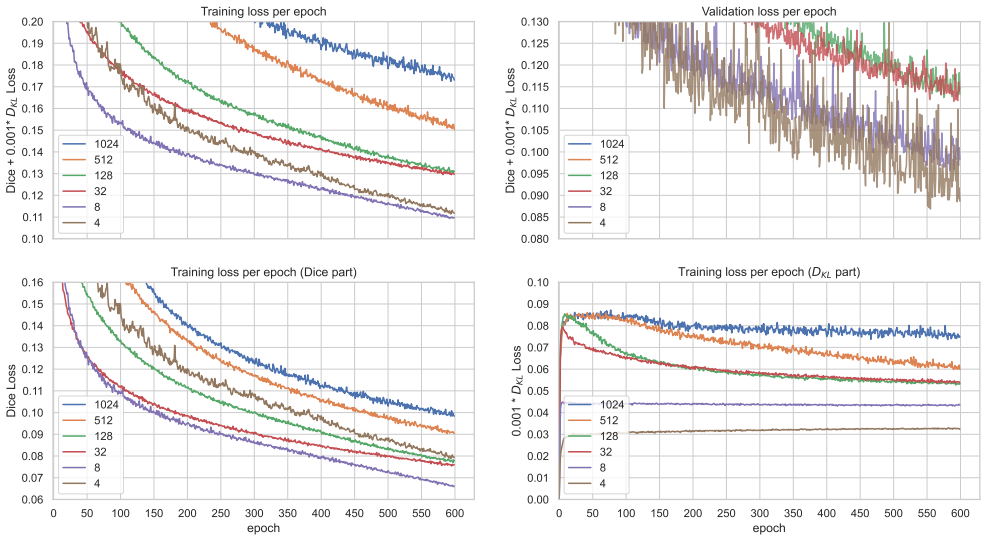

Figure 7: Comparison of average training and validation loss across all batches per epoch for a selection of latent space sizes. The diagrams in the bottom row showcase the dice and D_{KL} part of the training loss. Note: The latent space sizes 1024 and 512 are listed in the legend of the validation loss plot for completeness but are not visible due to worse results beyond the limit of 0.13.

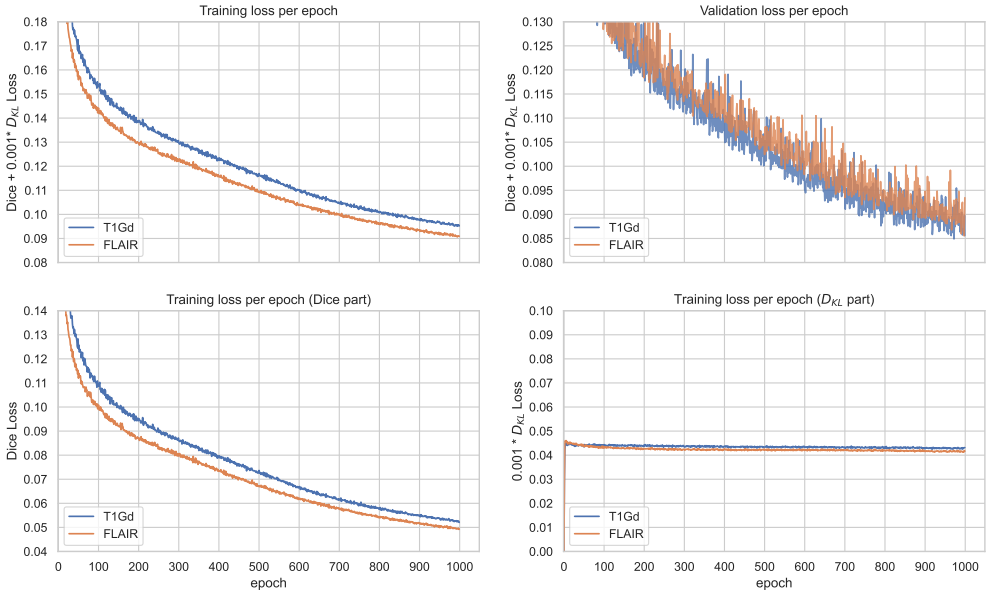

Figure 8: Comparison of average training and validation loss across all batches per epoch for the final architecture trained on the T1Gd and FLAIR segmentation. The diagrams in the bottom row showcase the dice and D_{KL} part of the training loss.

4 Evaluation

As a basis for our approach evaluation, we first assess in subsection 4.1.1 the quality of the baseline best-match. Each of the 62 real tumors $\in P$ is compared to the database of 50,000 synthetic tumors. For all comparisons, we used the dice metric implemented in [MON20]. The resulting dice scores for both segmentations and the combined values are ranked in 3 lists, yielding the rankings R^{T1Gd}, R^{FLAIR} and $R^{Combined}$.
Based on these dice scores and rankings, we assess in section 4.1 the different compression methods with respect to the preservation of the ranking as well as runtime improvements. In section 4.2 we evaluate the quality of the baseline and its optimizations by examining results for a selected subset of tumors and assess properties specific to the AE and VAE approaches. Finally, we investigate the embedding of our approach in the learn-morph-infer pipeline in section 4.3.

4.1 Quantitative Evaluation

4.1.1 Baseline

Before testing the different optimizations, we first have to assess the quality of the baseline results. For this purpose, we register the input tumors to the Atlas space using the ANTS framework [ATS+09] and then morph them utilizing the provided forward transformation. Now that both real tumors and synthetic tumors share the same brain anatomy, we can execute our query into a dataset of 50,000 synthetic tumors, yielding the best match for each input. The queries were executed utilizing 32 processes on a server hosting 128 AMD EPYC 7452 32-Core CPUs, taking an average of 340 seconds per query.
In Figure 9 we examine the dice scores between the pairs of real tumor and best synthetic match in the atlas space. This analysis suggests that the current dataset is in general not sufficient to extract a high quality match for real tumors. Although a few synthetic tumors achieve promising combined dice scores of roughly 1.4, even these best matches are probably too inaccurate to provide meaningful information in a clinical setting. Moreover, a significant amount of tumors has no decent match in the current database. This could have multiple reasons, such as e.g. a too simplistic underlying tumor model, an unusual tumor that was diagnosed early or late, an insufficient size of the database, etc. However, we still investigate this approach further, since the underlying dataset can be improved, e.g. by adding more data points or using a more complex and accurate tumor model. For instance, the currently used reaction diffusion model could be replaced with the brain deformation model, which is already implemented in the same glioma solver tool [LE20][Lip+19]. While, due to the concept of the pipeline, a change in the tumor model would not affect the runtime of our

approach, an extension of the dataset would increase the runtime linearly. The insufficiency of the current database therefore additionally motivates optimizations of the suggested query approach.

Figure 9: Dice scores (T1Gd, FLAIR and combined) between the morphed real input tumors $\in P$ in the atlas space and their best combined synthetic match. The average dice score $\overline{Dice}$ is indicated by the blue horizontal line ($\overline{Dice}_{T1Gd} = 0.522; \overline{Dice}_{FLAIR} = 0.549; \overline{Dice}_{Combined} = 1.072$).

4.1.2 Compression

The main goal of compression techniques in our context is to reduce the runtime of the database query, which will be evaluated in the following. However, since all tested methods are lossy, we also have to examine the quality and soundness of the compressions. For this purpose, we evaluated different metrics: Intersection of the top 15 encoded and ground truth results, ranking position of the best encoded match in the ground truth ranking $R^{Combined}$ and presence of the best match in the top n elements of $R^{Combined}$.
In a theoretical transfer to clinical practice, our proposed query would return the best match of the comparison. Therefore, this best match should be as highly ranked as possible in $R^{Combined}$, for which the last proposed metric seems the most significant. Consequently, we examine for each technique if the result is present in the top 1, 5 and 15 elements of $R^{Combined}$.

Down sampling

Both down sampling variations achieve a significant runtime improvement: The 64^3 version takes an average of 130 seconds per query and thereby only approximately 38% of the baseline. A down sampling to 32^3 shortens the runtime further, taking an average of 116 seconds per query. Figure 10 depicts the quality of the down-sampled results. As expected, the less compressed 64^3 variant manages to preserve the best match more frequently and also performs better in the top 5 comparison. Both versions are capable to return a tumor that was at least in the 15 closest matches in $R^{Combined}$. However, the stronger 32^3 compression is not able to preserve the best match for roughly a third of the test set, indicating that further simplistic compression will significantly worsen the results. To further compress the data or improve the quality of the down sampling, a more sophisticated interpolation method might be beneficial. Due to the fact that we use binary volumes, the interpolation improvements are however limited.

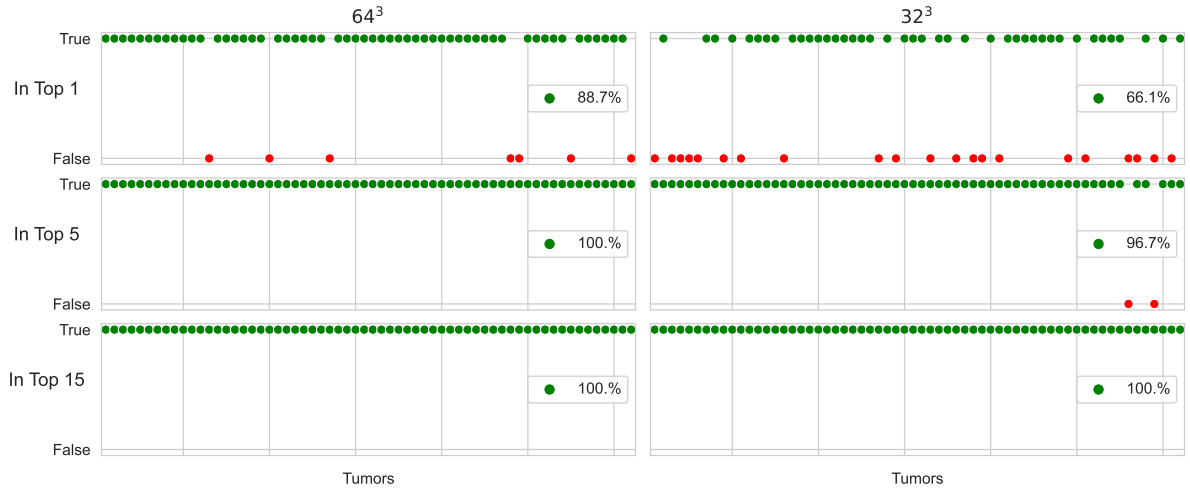

Figure 10: Plots showcasing the presence of the best down sampled match in the top n elements of $R^{Combined}$ for each of the 62 morphed patient tumors. The subplots display all combinations of the down sampling versions (64^3 and 32^3) and the top n gradations, as well as the percentage of *True* values.

Autoencoder

Using the autoencoder approach, a single query takes on average approximately 21 seconds. The majority of this time is spent loading the T1Gd and FLAIR networks and encoding the data, while the actual parallelized comparison only takes about 1.3 seconds (executed on the same hardware as in subsection 4.1.1). Figure 11 shows the preservation of the best match after encoding the data using the autoencoders from subsection 3.4.2. A third of the best matches can be preserved and 77.4% of the best matches are at least in the top 15 of $R^{Combined}$. While it is possible that the similarity loss is due to the significant compression, the results

appear quite bad when compared to the low validation loss from training. Therefore, we further investigated the cause of this loss in similarity, to ensure there is no generalization error that is not detected by the validation set. For this purpose, we analyzed the dice scores between input tumor and the respective reconstruction for the entire datasets S and P, and visualized them in Figure 12. For the synthetic data S, the results match the observed validation scores: For T1Gd the difference between validation score and average dice score for the entire dataset $|\overline{Dice}_{Validation} - \overline{Dice}_S|$ accounts to roughly 0.00055 and for FLAIR to approximately 0.00019. However, Figure 12 also shows that reconstruction losses for the real tumors are notably worse. This can be seen as a result of the shortcomings of S mentioned in subsection 4.1.1 and also offers an explanation for the similarity loss: The autoencoder struggles to extract meaningful features of the tumors in P that would allow a reconstruction, leading to a significant loss in similarity when comparing the encoded data during the query. While an improved dataset would certainly still yield a notable similarity loss, this analysis suggests that a better generalization for real tumors might improve the preservation of similarity relations and thereby also the performance of this optimization.

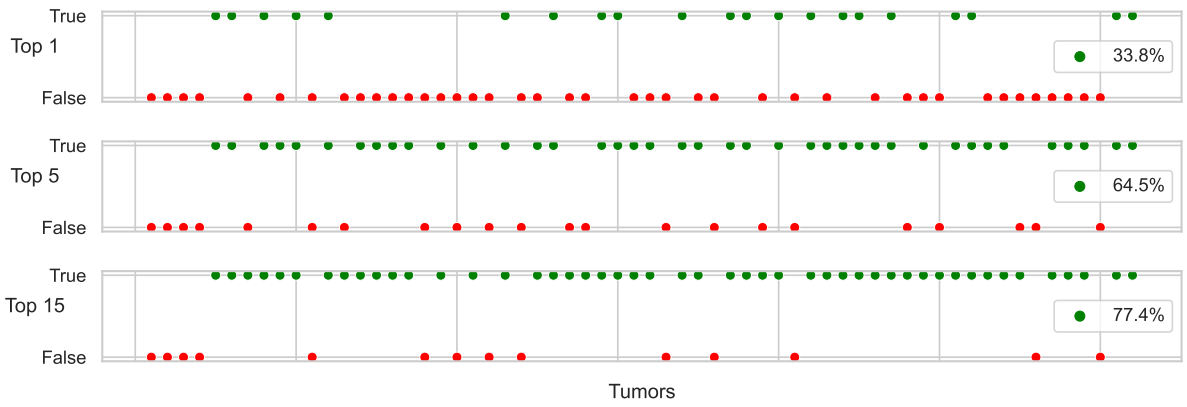

Figure 11: Plots showcasing the presence of the best AE encoded match in the top n elements of $R^{Combined}$ for each of the 62 morphed patient tumors. The subplots depict the categories top 1, 5 and 15 and for each the percentage of *True* values.

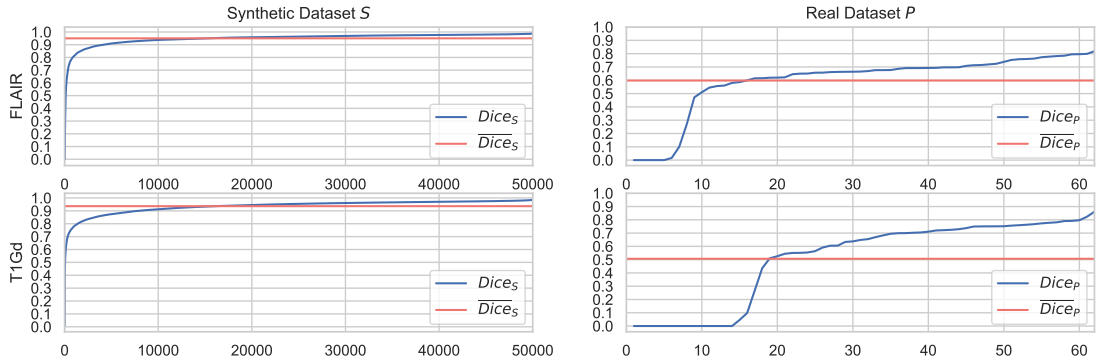

Figure 12: Diagrams depicting the dice score between the input tumor and its AE reconstruction for both segmentations (T1Gd and FLAIR) and datasets (S and P). As a preprocessing step, all results have been sorted by dice score. The average dice score $\overline{Dice}$ has been plotted for all combinations.
($\overline{Dice}_S^{FLAIR} = 0.950; \overline{Dice}_S^{T1Gd} = 0.936; \overline{Dice}_P^{FLAIR} = 0.598; \overline{Dice}_P^{T1Gd} = 0.507$).

Variational Autoencoder

The VAE approach provides a further runtime improvement. A single query returns a result within roughly 7 seconds, when executed on the same hardware as in the other sections. Similar to the autoencoder approach, the majority of this time is spent loading the both networks and encoding the data, while the actual best match calculation only accounts for 1.3 seconds. The following assessment resembles the previous autoencoder evaluation. Figure 13 shows the preservation of the top n best matches after encoding S using the variational autoencoder models from subsection 3.4.3. In a direct comparison, the VAE approach performs worse in most categories. It is only capable to preserve the best match in 19.3% of the tests, and also the top 15 preservation drops from 77.4% for the AE approach to 67.7% for the variational autoencoder. Figure 14 indicates that this performance loss could be explained with a worse transferability of the VAE networks from synthetic to real data and an overall worse reconstruction performance that could limit the ability to encode relevant properties. While the drop of average dice scores $\overline{Dice}$ to 0.902 for FLAIR and 0.881 for T1Gd between input and reconstruction for elements of S is tolerable ($AE : \overline{Dice}_{FLAIR} = 0.950; \overline{Dice}_{T1Gd} = 0.936$), the worse reconstructions for the tumors of P ($VAE : \overline{Dice}_{FLAIR} = 0.526; \overline{Dice}_{T1Gd} = 0.438; AE : \overline{Dice}_{FLAIR} = 0.598; \overline{Dice}_{T1Gd} = 0.507$) seem to negatively influence the similarity preservation. Although the preceding results render the employability of the VAE approach questionable, it has to be stated that the ability to preserve the best match in roughly 19% of the test cases out of 50,000 possible options after compressing the data from 128^3 to 8 values (a compression of factor 2^{18}), manifests that it is possible to encode problem specific similarity relations at least to a certain degree. The top 15 result of 67.7% confirms this.

	Baseline	DS 64	DS 32	AE	VAE 1024	VAE 8	VAE 8 20K
Top 1	/	88.7%	66.1%	33.8%	41.9%	19.3%	30.6%
Top 5	/	100%	96.7%	64.5%	61.2%	48.3%	56.4%
Top 15	/	100%	100%	77.4%	80.6%	67.7%	69.3%
Runtime	340s	130s	116s	22s	34s	7s	7s

Table 1: Table summarizing the results of section 4.1. The baseline and all optimizations are compared w.r.t the preservation of the top n matches for the test data and the runtime per individual query in seconds. The column headers are abbreviations of the approaches: Down sampling to resolution X (DS X), autoencoder (AE) and variational autoencoder with latent size X and optionally training set size T (VAE X [T]).

Regardless of the worse training results, we additionally evaluated a VAE with latent size 1024 to have a fair comparison with our autoencoder approach. Although the reconstructions of the larger latent space are significantly worse (see Figure 7), this configuration outperforms the smaller latent size and even the autoencoder approach in terms of the best match preservation. The second column in Figure 13 demonstrates this, suggesting that the reconstruction loss alone can not be used to optimize similarity preservation.
Motivated by this result, we experimented with other parameters that did not improve training results but might prove beneficial for similarity preservation. For this purpose, we trained a VAE with latent space size 8 with a training set of 20,000 synthetic tumors (the bigger latent size of 1024 would not have been able to finish training within the timeframe of the project) for 120 epochs, yielding a total training time of approximately 204 hours. The resulting model for the T1Gd modality achieved a training loss of roughly 0.103, a pure dice score of 0.059 and a validation loss of 0.116 (smaller training set: training loss = 0.952, pure dice loss = 0.052, validation loss = 0.085). Although the raw values are slightly worse for the larger training set, Figure 13 shows that the best match preservation benefits from it, which could be explained by a better generalization due to an increased input variety.
Note: Since these improvements were found late in the project, all following evaluations and visualizations utilize the VAE with latent size 8 and training set size 1500. However, since the adaptions provide promising results, they should be further investigated in future research.

Table 1 compares the presented baseline and all discussed optimizations. As expected, the optimizations significantly improve runtime while losing accuracy in terms of similarity preservation. Given the tolerable runtime of the baseline for the current database, this basic approach should be used if accuracy is the most important property, which typically applies in a clinical setting. However, if the database should significantly increase in size, the proposed optimizations can be utilized. The choice of optimizations depends on the needed level of compression.

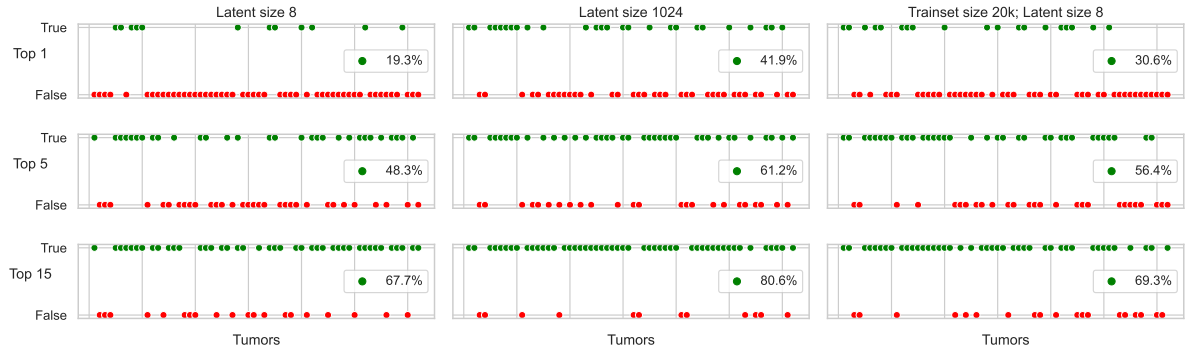

Figure 13: Plots showcasing the presence of the best VAE encoded match in the top n elements of $R^{Combined}$ for each of the 62 morphed patient tumors. The subplots depict the categories top 1, 5 and 15 for different models. Two models use a training set size of 1,500 with latent sizes 8 and 1024, while the third one uses a training set size of 20,000 with latent size 8. For each top-n-model combination, the percentage of *True* values is displayed.

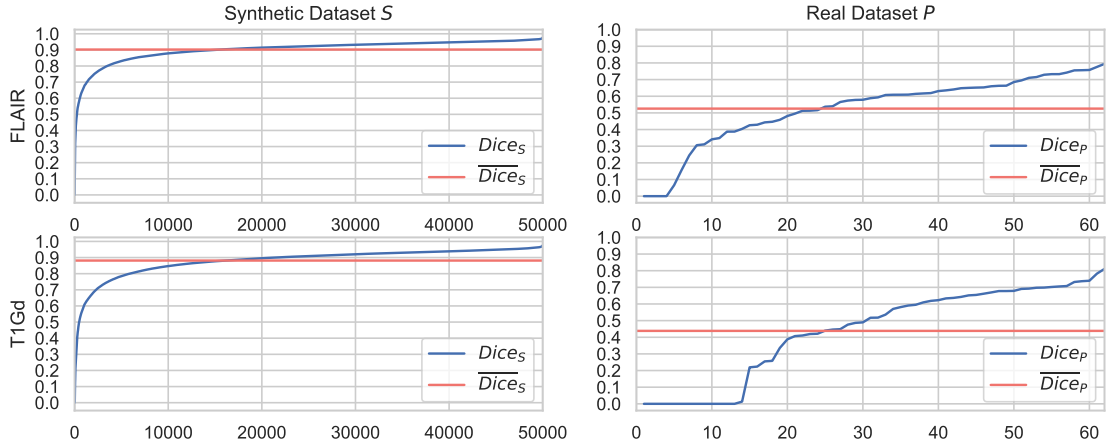

Figure 14: Diagrams depicting the dice score between the input tumor and its VAE reconstruction for both segmentations (T1Gd and FLAIR) and datasets (S and P). As a preprocessing step, all results have been sorted by dice score. The average dice score $\overline{Dice}$ has been plotted for all combinations.
($\overline{Dice}_S^{FLAIR} = 0.902; \overline{Dice}_S^{T1Gd} = 0.881; \overline{Dice}_P^{FLAIR} = 0.526; \overline{Dice}_P^{T1Gd} = 0.438$).

4.2 Qualitative Evaluation

We evaluate the quality of our approach for a selected subset T of 5 tumors $\subset P$. The individual brain tumors were selected to provide an overview of the capabilities as well as the shortcomings of the base approach and its optimizations. To achieve this goal, we split the dataset P into 5 sections based on the performance in Figure 9, resulting in the set of evenly sized ranges $\{[0.5, 0.7]; [0.7, 0.9]; [0.9, 1.1]; [1.1, 1.3]; [1.3, 1.5]\}$. The lower bound is motivated by the lack of useful information of results below it, the upper bound is selected since no tumor performed better. From these ranges, we picked tumors, while trying to keep the preservation rate of the best match of each optimization method as close as possible to the average quantitative results from section 4.2. The final selection is decreasingly ordered by the combined dice score and depicted in Table 2.
Note: Due to the limited size of P not all constraints can be met, however we argue that the chosen subset still provides useful insights.
In subsection 4.2.1 we analyze the qualitative results of the baseline and its optimizations for the tumor subset and in subsection 4.2.2 we evaluate properties specific to the autoencoder and VAE approach.

4.2.1 General

In the following, we will take a closer look at the individual quality of our approach for the previously selected subset of tumors by investigating the query results. Figure 15 visualizes the morphed segmentations of the input tumors along with their best match from the database embedded in the atlas brain anatomy. To reduce clutter, the best matches of the down sampling approaches are not depicted, since they are mostly equal to the baseline. The figure shows that all best matches return tumors that are at least roughly similar to the input and visually even closer to the baseline result, confirming the similarity preserving property of the optimizations. Moreover, the plots also indicate that seemingly unpromising rankings, e.g. the AE (rank 26) and VAE (rank 23) results of tumor 5, can still be quite close to the best match of the baseline.
Furthermore, it can be seen that the synthetically generated tumors have much smoother shapes in comparison to the real inputs. While this might be a shortcoming of the input scans on the one hand, it certainly also renders the realism of the synthetic tumors questionable on the other. Additionally, the query result's T1Gd to FLAIR ratio is far higher than for the real tumor scans, especially for matches with worse dice scores. Thus, an increase of the FLAIR modality's weighting in the calculation of the combined dice score during the query could improve results.

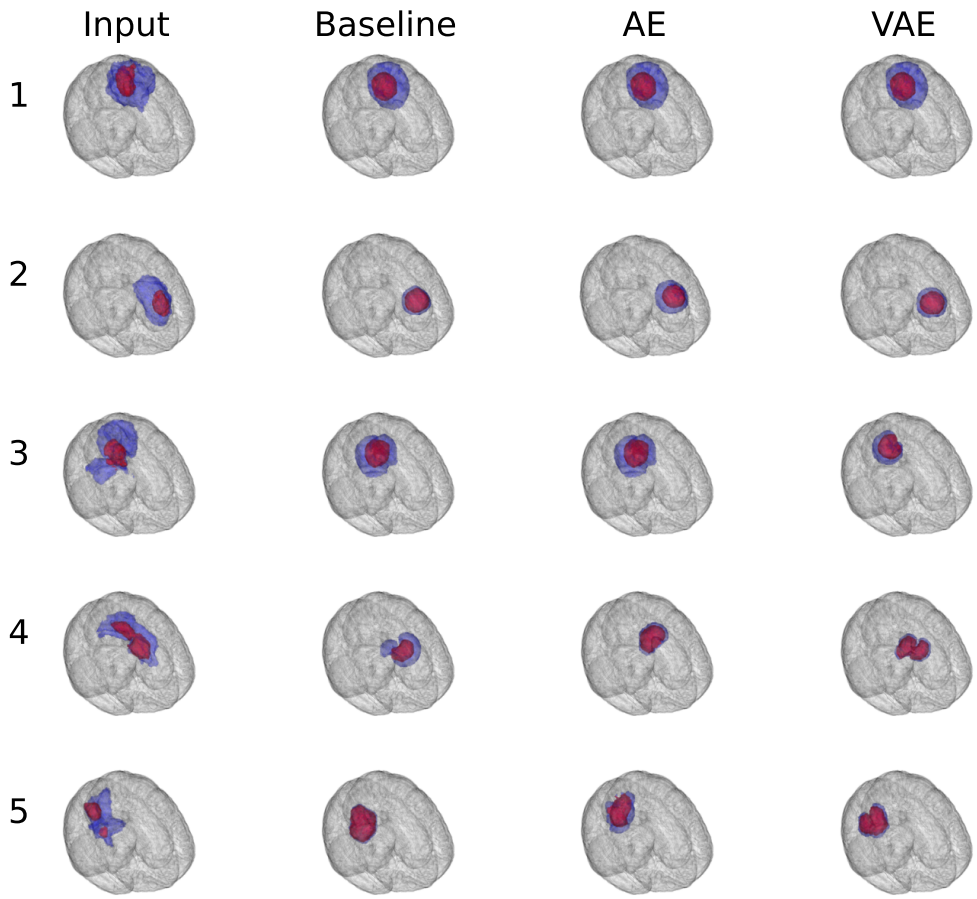

Figure 15: Visualizations of the segmentations of the input tumor as well as the best match of the baseline, the autoencoder (AE) and the variational autoencoder (VAE) embedded in the atlas brain anatomy for all tumors $\in T$ (FLAIR in blue, T1Gd in red).

Tumor	1 (047)	2 (008)	3 (051)	4 (025)	5 (023)
Combined	1.389	1.201	0.968	0.811	0.683
T1Gd	0.646	0.635	0.561	0.428	0.155
FLAIR	0.743	0.566	0.407	0.383	0.527
DS 64	1 (Δ=0)	1 (Δ=0)	1 (Δ=0)	1 (Δ=0)	2 (Δ=0.02)
DS 32	1 (Δ=0)	1 (Δ=0)	1 (Δ=0)	1 (Δ=0)	1 (Δ=0)
AE	1 (Δ=0)	7 (Δ=0.09)	1 (Δ=0)	43 (Δ=0.20)	26 (Δ=0.13)
VAE	1 (Δ=0)	3 (Δ=0.07)	2 (Δ=0.16)	3 (Δ=0.02)	23 (Δ=0.12)

Table 2: Overview of relevant properties of the selected tumors. Combined, T1Gd and FLAIR are the respective dice scores from Figure 9. The values in the rows DS 64 (down sampling to 64^3), DS 32 (down sampling to 32^3), AE (autoencoder) and VAE (variational autoencoder) are the ranking of the given optimization's best match in the ground truth ranking $R^{Combined}$ (1 meaning the same best match). Additionally, the difference Δ between the combined dice scores of the best match of the baseline and the respective optimizations are shown.

4.2.2 Autoencoder & Variational Autoencoder

In this section, we shortly evaluate the quality of the autoencoder and VAE networks with respect to compression and reconstruction capabilities.
The final AE architecture compresses the data from a resolution of 128^3 to 1024 values, yielding a compression rate of 2^{11}, while achieving high average reconstruction dice scores for synthetic tumors, as visualized in Figure 12. These results are qualitatively manifested in Figure 16 for 3 example tumors, that showcase a close reconstruction even though the chosen tumors perform below average. As already discussed in section 4.1.2 the performance drops significantly when applying the network to real data, which is also depicted in Figure 17.
Our variational autoencoder achieves an even higher compression rate of 2^{18} (from 128^3 to 8 values). However, this comes at the cost of worse reconstruction results, as can be seen for the entire datasets in Figure 14 as well as qualitatively in Figure 16 and Figure 17. Considering the notable compression difference, the slight losses in reconstruction accuracy might be tolerable, if the main focus is compression.
Reconstruction results of real tumors for both autoencoder types suffer from the smoothness of the synthetic training data. Since all synthetic tumors have rather smooth surfaces, as discussed in subsection 4.2.1, the rough input shapes could be a reason for the worse reconstructions. This further motivates the improvement of the underlying dataset.

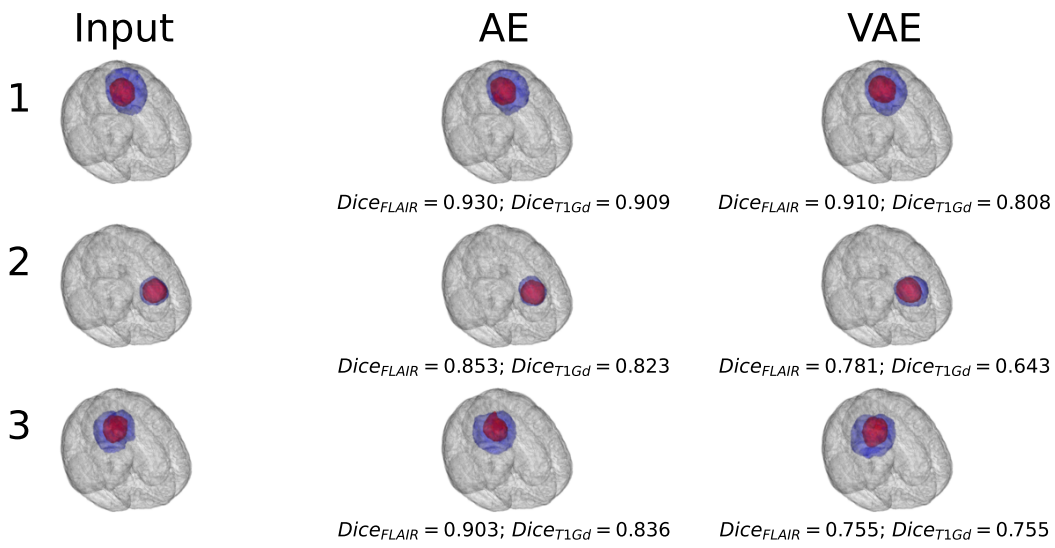

Figure 16: Visualizations of the segmentations of the synthetic input tumor and its reconstructions using the autoencoder (AE) and the variational autoencoder (VAE) approach embedded in the atlas brain anatomy (FLAIR in blue, T1Gd in red). The shown tumors are the baseline best matches of the elements 1 to 3 of the selected subset T. The FLAIR and T1Gd dice scores between reconstruction and input are depicted below the AE and VAE images.

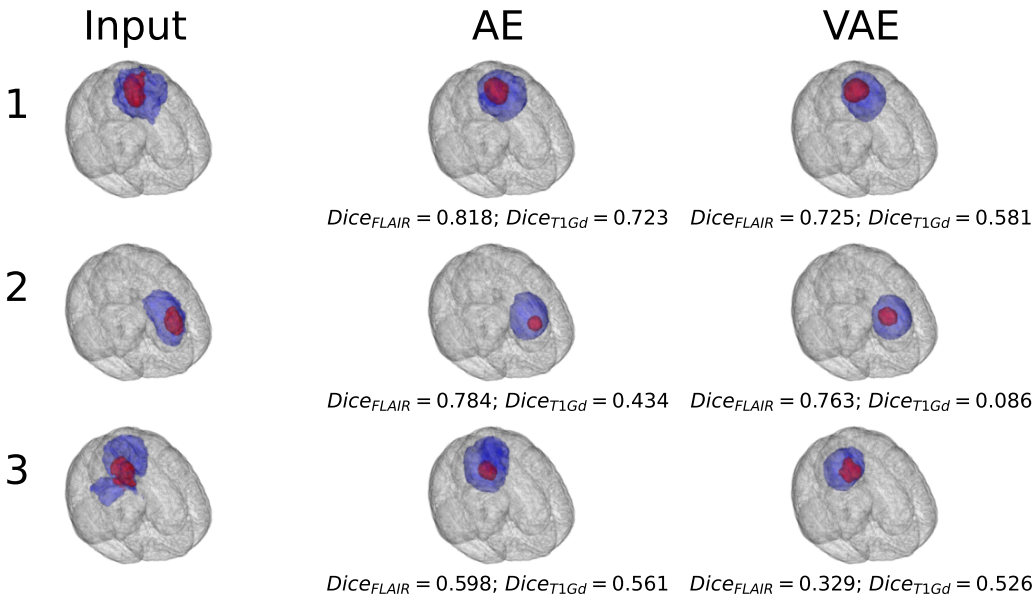

Figure 17: Visualizations of the segmentations of the real input tumor and its reconstructions using the autoencoder (AE) and the variational autoencoder (VAE) approach embedded in the atlas brain anatomy (FLAIR in blue, T1Gd in red). The shown tumors are the elements 1 to 3 of the selected subset T. The FLAIR and T1Gd dice scores between reconstruction and input are depicted below the AE and VAE images.

4.3 Performance in the Pipeline Context

Finally, we evaluate the integration of our proposed approach in the learn-morph-infer pipeline. To recap:
We first register the patient brain to the atlas brain and then morph it using the provided forward registration transformation. Afterwards, we execute two individual queries into the database S, yielding dice scores for the FLAIR and T1Gd modalities. The highest combined score is selected as the best match and then morphed back to the patient brain space, obtaining the final result.
These steps were implemented and executed utilizing the baseline approach and the 5 test tumors from section 4.2 as input. The results are visualized in Figure 18. The first tumor provides a satisfying result. As expected, due to the selection of tumors by descending dice scores, the remaining results progressively decrease in quality. The last tumor can not be visualized at all, since it has no data in the given brain slice. However, this does not mean that it is not at all similar to the real tumor, as can be seen in Figure 15. Still, this fortifies the earlier discussed need of a more accurate and richer dataset in order to make the proposed approach applicable.

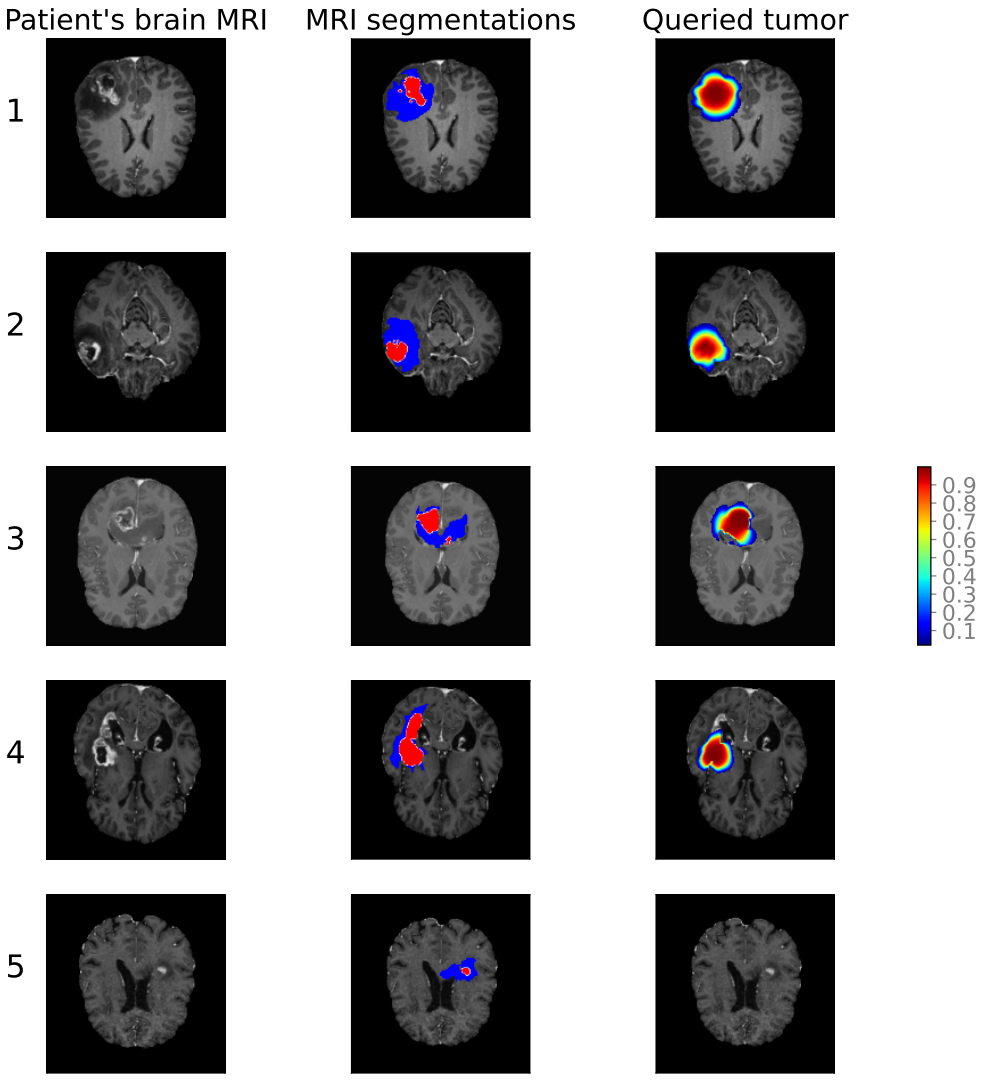

Figure 18: Visualization of the final pipeline results. The 5 selected tumors are displayed as raw MRI in the first column, continued with a highlight of the MRI segmentations (FLAIR in blue, T1Gd in red) in column 2. The final column shows the patient brain anatomy specific tumor cell density of the queried tumor in the given brain slice. All visualizations of the same tumor are showing the same slice, which is determined by the maximum of non-zero voxels per scan.

5 Future Work

Throughout this work we have seen that even for the fully deterministic and simplistic baseline, our approach is with the current database generally not sufficient to extract a high quality match. Therefore, the most important future improvement is the enhancement of the underlying synthetic dataset. This could be achieved by using a different tumor model. The currently used reaction diffusion model utilizes the relatively simple fisher-kolmogorov partial differential equation [Lip+19]. In order to hopefully generate more realistic synthetic tumors, this could be replaced with the brain deformation model that additionally considers mass effect and intracranial pressure [LE20]. Due to the concept of the learn-morph-infer pipeline, a more sophisticated tumor model would not affect the runtime of the approach [Ezh+21b]. Furthermore, the size of the dataset could be increased as well to provide closer matches for the broad variety of real tumor inputs. An extension of the evaluation dataset of real tumors P would also be beneficial to increase confidence in possible optimizations.
The different compression methods in this work can roughly be seen as a trade-off between accuracy and efficiency. However, we believe that the currently implemented optimizations can be improved to achieve less similarity loss while still significantly compressing the data. For both the autoencoder and variational autoencoder approaches, the chosen architecture and hyperparameters may not be optimal due to the time constraints of this project. Therefore, further experiments with different convolutional and linear layers as well as the number of channels or an adaptive learning rate might prove beneficial. Moreover, the improvements discussed in the VAE part of subsection 4.1.2 could be further investigated. Additionally, the proposed methods could be concatenated to combine their strengths. For instance, down sampling could be used as a preprocessing step before feeding the data to the neural networks.

Finally, other approaches to optimize the baseline approach could be examined. For example, [BP17] introduce a compression method for binary vectors that preserves the hamming distance and inner product. They generalize their approach to also compress real valued vectors while preserving the euclidean distance and propose a potentially efficient nearest neighbor search that first compresses the data using their approach and then applies a collision based hashing algorithm. While sounding promising, it has to be noted that this method only works for vectors and is therefore requiring a flattening of the volumes in our application. Thereby, the similarity preservation of the compressed data may be better but less useful since the underlying metric is not as meaningful for our use case as the dice metric, which is not applicable for vectors. Another option could be similarity preserving hashing functions, for which [MÁE14] presents an overview of potential methods.

6 Conclusion

We have shown that it is possible to utilize an image retrieval based approach as a deterministic replacement for parameter inferring neural networks in the learn-morph-infer pipeline. While the currently used database is not sufficient to provide a close match in general, we argue that the performance of our approach can be significantly improved with a more sophisticated dataset.
Furthermore, we manifested that deep learning based encodings can preserve problem-specific similarity relations up to a certain degree while drastically reducing data size and runtime. We believe that, considering the improvement of tumor modelling accuracy and complexity as well as the constantly growing masses of data, our results can be useful in future research.

Bibliography

[AC17] Brian M. Alexander and Timothy F. Cloughesy. "Adult Glioblastoma". In: *Journal of Clinical Oncology* 35.21 (2017). PMID: 28640706, pp. 2402–2409. DOI: 10.1200/JCO.2017.73.0119.

[ATS+09] Brian B Avants, Nick Tustison, Gang Song, et al. "Advanced normalization tools (ANTS)". In: *Insight j* 2.365 (2009), pp. 1–35.

[BP17] Raghav Kulkarni LinkedIn Bangalore and Rameshwar Pratap. *Workshop track-ICLR 2017 Similarity Preserving Compressions Of High Dimensional Sparse Data*. 2017.

[Ezh+21a] Ivan Ezhov et al. *Geometry-aware neural solver for fast Bayesian calibration of brain tumor models*. 2021. arXiv: 2009.04240 [cs.CE].

[Ezh+21b] Ivan Ezhov et al. *Learn-Morph-Infer: a new way of solving the inverse problem for brain tumor modeling*. 2021.

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016. Chap. 14.

[Hig+] Irina Higgins et al. *-VAE: Learning Basic Visual Concepts with a Constrained Variational Framework*.

[JDJ17] Jeff Johnson, Matthijs Douze, and Hervé Jégou. "Billion-scale similarity search with GPUs". In: *arXiv preprint arXiv:1702.08734* (2017).

[KB17] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].

[KW13] Diederik P Kingma and Max Welling. "Auto-Encoding Variational Bayes". In: (Dec. 2013). URL: http://arxiv.org/abs/1312.6114.

[Lê+16] Matthieu Lê et al. "Planning Based on a Computational Tumor Growth Model". In: *IEEE Transactions on Medical Imaging, Institute of Electrical and Electronics Engineers* (2016). DOI: 10.1109/TMI.2016.2626443Ãŕ. URL: https://hal.inria.fr/hal-01403847.

[LE20] Jana Lipkova and Ivan Ezhov. *Software for realistic tumor modeling*. 2020. URL: https://github.com/IvanEz/GliomaSolver.

[Lip+19] Jana Lipkova et al. "Personalized Radiotherapy Design for Glioblastoma: Integrating Mathematical Tumor Models, Multimodal Scans, and Bayesian Inference". In: *IEEE Transactions on Medical Imaging* 38 (8 Aug. 2019), pp. 1875–1884. ISSN: 1558254X. DOI: 10.1109/TMI.2019.2902044.

[MÁE14] Víctor Gayoso Martínez, Fernando Hernández Álvarez, and Luis Hernández Encinas. *State of the Art in Similarity Preserving Hashing Functions*. 2014.

[Men15] Bjoern H. et al. Menze. "The Multimodal Brain Tumor Image Segmentation Benchmark (BRATS)". In: *IEEE Transactions on Medical Imaging* 34.10 (2015), pp. 1993–2024. DOI: 10.1109/TMI.2014.2377694.

[MON20] MONAI Consortium. *Project MONAI*. 2020.

[Roh+10] Torsten Rohlfing et al. "The SRI24 multichannel atlas of normal adult human brain structure". In: *Human Brain Mapping* 31.5 (2010), pp. 798–819. DOI: https://doi.org/10.1002/hbm.20906.

[RV11] P. Ramachandran and G. Varoquaux. "Mayavi: 3D Visualization of Scientific Data". In: *Computing in Science & Engineering* 13.2 (2011), pp. 40–51. ISSN: 1521-9615.

[Sci21] Kevin Robert Scibilia. *Learn-Morph-Infer: A New Paradigm of Brain Tumor Growth Model Personalization (a Deterministic Scenario)*. B.Sc. thesis. Technical University of Munich, 2021.

[Ste20] Felix Steinbauer. *Learn-Morph-Infer: A Novel Paradigm of Brain Tumor Model Calibration*. B.Sc. thesis. Technical University of Munich, 2020.

[Stu+14] R. Stupp et al. "High-grade glioma: ESMO clinical practice guidelines for diagnosis, treatment and follow-up". In: *Annals of Oncology* 25 (Sept. 2014), pp. 93–101. ISSN: 15698041. DOI: 10.1093/annonc/mdu050.

[TH15] Abdel Aziz Taha and Allan Hanbury. "Metrics for evaluating 3D medical image segmentation: Analysis, selection, and tool". In: *BMC Medical Imaging* 15 (1 Aug. 2015). ISSN: 14712342. DOI: 10.1186/s12880-015-0068-x.

[Vir+20] Pauli Virtanen et al. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17 (2020), pp. 261–272. DOI: 10.1038/s41592-019-0686-2.