DeepBrainPrint: A Novel Contrastive Framework for Brain-MRI Re-Identification

Author name(s) withheld EMAIL(S) WITHHELD
Address withheld

Editors: Under Review for MIDL 2023

Abstract

Recent advances in MRI have led to the creation of large datasets. With the increase in data volume, it has become difficult to locate previous scans of the same patient within these datasets (a process known as re-identification). To address this issue, we propose an AI-powered medical imaging retrieval framework called DeepBrainPrint, which is designed to retrieve brain MRI scans of the same patient. Our framework is a semi-self-supervised contrastive deep learning approach with three main innovations. First, we use a combination of self-supervised and supervised paradigms to create an effective brain fingerprint from MRI scans that can be used for real-time image retrieval. Second, we use a special weighting function to guide the training and improve model convergence. Third, we introduce new imaging transformations to improve retrieval robustness in the presence of intensity variations (i.e. different scan contrasts), and to account for age and disease progression in patients. We tested DeepBrainPrint on a large dataset of T1-weighted brain MRIs from the Alzheimer's Disease Neuroimaging Initiative (ADNI) and on a synthetic dataset designed to evaluate retrieval performance with different image modalities. Our results show that DeepBrainPrint outperforms previous methods, including simple similarity metrics and more advanced contrastive deep learning frameworks.

Keywords: Contrastive Learning, Brain MRI Fingerprint, Re-Identification

1. Introduction

The demand for radiologists to interpret brain MRIs is increasing due to the growing number of MRI scans and advances in MRI technology. To alleviate the burden on hospitals, tools for efficient image retrieval can be crucial. Retrieving scans from the same subject can provide valuable insights for analyzing disease progression over time. However, it is important to consider privacy when developing these tools (i.e. avoiding the use of the subject's name).

Metadata, such as subject IDs, could be useful for this, but their referential integrity may not always be reliable. This is because different IDs may be assigned to the same subject within the same hospital or across different hospitals. To address this issue, our work focuses on re-identifying brain MRIs using image content directly.

While several solutions exist for this purpose, all of them have limitations. For example, facial recognition algorithms can be used on MRIs (Schwarz et al., 2019), but faces and skulls are often removed during the anonymization process, making this solution impractical.

Other approaches involve aligning MRI into a common template and comparing them using image similarity based on intensity values (e.g. Minimum Cross Entropy or Structural Similarity Index) (Al-bory and Elsheh, 2018). However, the number of voxels is often too large to evaluate image similarity in real-time for each new query. Additionally, each voxel

© 2023 Anonymous.

does not carry anatomic information that characterizes the brain, leading to the retrieval of images that do not necessarily represent the same anatomical phenotype (Faria et al., 2015). To overcome this, (Wachinger et al., 2015) proposed a solution that captures geometric information by using the spectrum of the Laplace-Beltrami operator on meshes derived from brain MRIs, making the approach robust to intensity variations. However, even this method is computationally intensive and requires segmentation, hindering real-time processing.

Alternative solutions for fast, real-time retrieval are based on the recent progress of deep metric learning. These approaches develop efficient representation functions that avoid heavy processing (e.g., segmentation) when querying a new scan. These functions map each scan into an embedding space, ensuring that the distances in the new manifold reflect the initial similarity between scans. Approaches based on deep metric learning can be divided into two main categories: fully-supervised and self-supervised.

In the fully-supervised category, we found deep Siamese networks (Chopra et al., 2005) which use labelled data to train a contrastive loss (Kumar BG et al., 2016) such as Triplet (Hoffer and Ailon, 2015), SoftTriple (Qian et al., 2019), Proxy-NCA (Movshovitz-Attias et al., 2017), and Normalized Temperature-scaled Cross Entropy (Chen et al., 2020)). However, recent research (Musgrave et al., 2020) has shown that these supervised contrastive solutions have reached a limit in terms of improvements.

Unlike fully-supervised approaches, self-supervised approaches do not use labelled scans during training, but rather rely on transformations to create multiple distorted images that belong to the same subject (e.g., BarlowTwins (Zbontar et al., 2021), SimCLR (Chen et al., 2020)). While these self-supervised approaches have demonstrated impressive results on natural images, the transformations they use may not be suitable for MRIs. In addition, avoiding the use of available labels (i.e., subject IDs) could be a limitation in this task, as this information can be relevant to improving model performance.

In this work, we propose a hybrid solution that tries to overcome the limitations above. In particular, we extend the approach in (Zbontar et al., 2021) by adding three main novelties: i) we combine a self-supervised paradigm with a supervised paradigm to optimize a multi-task problem that leverages the availability of labels and improve retrieval performance; ii) we introduce a method to weight the importance of each task during training and improve model convergency; iii) we propose a novel set of image transformations designed specifically for brain MRIs and necessary to make the framework robust to various sources of variability. Our solution operates on a single 2D slice, which facilitates real-time retrieval.

To the best of our knowledge, this is the first semi-self-supervised approach for the retrieval of brain MRIs. Furthermore, our domain-specific image transformations are designed to make the retrieval robust to intensity changes, disease progression, and patient ageing, leading the framework to learn an adequate representation of brain morphology and achieving a recall close to 99% during retrieval.

2. Method

In this section, we provide the detail of our framework which include i) scan pre-processing, ii) training pipeline, iii) image transformations, and iv) training settings.

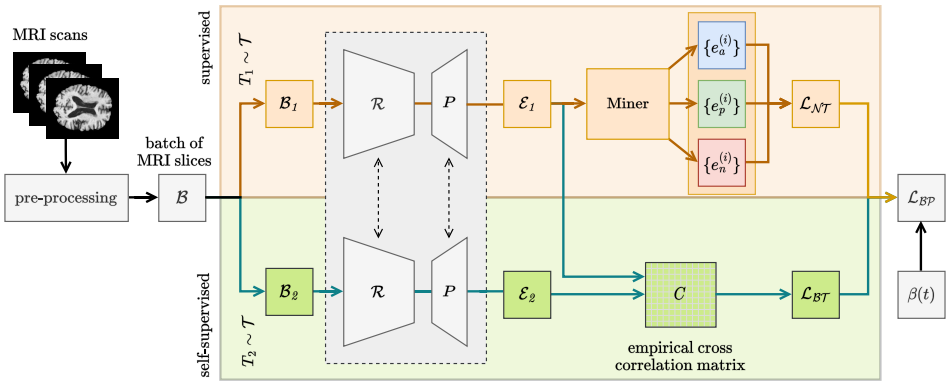

Figure 1: Workflow of our DeepBrainPrint pipeline which is divided into two main branches: a fully supervised branch (in orange) and a self-supervised branch (in green).

2.1. Pre-processing

This block aims to reduce critical but non-informative variations in the data and prepare scans for model training. Four sequential steps are used to process each scan: (i) co-linear registration to an MNI152 template (using the ANTs library), (ii) removal of 5% of intensity outliers from the entire scan, (iii) normalization of intensity using the z-score on each voxel, and (iv) extraction of the central 2D slice x from the axial plane.

2.2. Proposed DeepBrainPrint pipeline

The blocks of our training pipeline are depicted in Figure 1. The pipeline is divided into two main branches aimed to optimize two different properties of the embedding vectors: i) the self-supervised branch (in light green) ensures that all the representations obtained from scans from the same subject are similar to each other even if they are affected by some variability (e.g., different contrasts, different scanners, different protocols, etc.) and ii) the supervised contrastive branch (in light orange) ensures that representations obtained from follow-up scans from the same subject are similar to each other whereas scans from different subjects are dissimilar from each other. In particular, the self-supervised branch is inspired by the work proposed in (Zbontar et al., 2021) in which we integrate new domain-specific image transformations designed for brain MRI (see Section 2.3). The supervised branch, instead, follows a standard contrastive setting with triplets used to optimize the NT-Xent loss (Chen et al., 2020).

The training starts by sampling random batches $\mathcal{B}$ with size b containing a set of pre-processed slices $x_1, \ldots, x_b$. Then, for each 2D slice $x_i \in \mathcal{B}, i = 1, \ldots, b$ we sample two transformations $T_1, T_2 \sim \mathcal{T}$ (see Section 2.3) and obtain two distorted version $x_i' = T_1(x_i)$ and $x_i'' = T_2(x_i)$ of the scan x_i. By repeating this process for each slice x_i, we generate two sets of batches defined as:

$$\begin{aligned} \mathcal{B}_1 &= \{x_i' = T_1(x) \mid x_i \in \mathcal{B}, T_1 \sim \mathcal{T}\} \\ \mathcal{B}_2 &= \{x_i'' = T_2(x) \mid x_i \in \mathcal{B}, T_2 \sim \mathcal{T}\} \end{aligned} \tag{1}$$

These two batches are then passed to the encoder network $\mathcal{R}$, which shares weights between the two branches of the pipeline using a Siamese configuration. $\mathcal{R}$ maps each

transformed slice to a low-dimensional vector (the final "representation") having size z. Following this, a projection head P, having also shared weights, maps the representations obtained from $\mathcal{R}$ in vectors lying in a non-linear space with dimension s. We will refer to these new vectors as the "embeddings" e. Let's define the composition of the encoder and the projection head as $\mathcal{M}(\cdot) = P(\mathcal{R}(\cdot))$ and the corresponding embedding batches as $\mathcal{E}_1 = \mathcal{M}(\mathcal{B}_1)$ and $\mathcal{E}_2 = \mathcal{M}(\mathcal{B}_2)$. The pipeline proceeds by processing $\mathcal{E}_1$ and $\mathcal{E}_2$ and computing two main losses.

The self-supervised branch computes the loss $\mathcal{L}_{\mathcal{BT}}$ from (Zbontar et al., 2021), defined as follows:

$$\mathcal{L}_{\mathcal{BT}} = \sum_i (1 - c_{ii})^2 + \lambda \sum_i \sum_{j \neq i} c_{ij}^2; \tag{2}$$

where c_{ij} is an element of the cross-correlation matrix C, computed on $(\mathcal{E}_1, \mathcal{E}_2)$. Each c_{ij} indicates how much the i-th feature in the embedding vector is correlated to the j-th feature. The rightmost term called the "redundancy reduction term" ensures that the correlation between features is minimized, and the hyperparameter λ weights its final contribution in the loss. Instead, the left term, called the "invariance term", ensures that features extracted from different transformations of the same image are correlated, which makes the embedded representation invariant to the proposed image transformations.

On the other side, the supervised branch, which is trained simultaneously, computes the loss $\mathcal{L}_{\mathcal{NT}}$ which is inspired by (Chen et al., 2020) and defined as follows:

$$\mathcal{L}_{\mathcal{NT}} = -\log \frac{\exp\left(sim(e_a^{(i)}, e_p^{(i))})/\tau\right)}{\exp\left(sim(e_a^{(i)}, e_p^{(i))})/\tau\right) + \exp\left(sim(e_a^{(i)}, e_n^{(i))})/\tau\right)} \tag{3}$$

where $sim(\cdot, \cdot)$ is the cosine similarity function and $(e_a^{(i)}, e_p^{(i)}, e_n^{(i)})$ is a triplet generated by the mining block, which operates on the batch of embeddings $\mathcal{E}_1$. In particular, $e_a^{(i)}$ (the anchor sample) and $e_p^{(i)}$ (the positive sample) belong to the same subject, while $e_n^{(i)}$ (the negative sample) belongs to a different subject. The hyperparameter τ (called temperature) controls the magnitude of the similarities.

Finally, our pipeline optimizes the loss defined in Eq. 4, which combines $\mathcal{L}_{\mathcal{NT}}, \mathcal{L}_{\mathcal{BT}}$ by using a weighting function $\beta(t)$ that weights the contribution of each loss at each epoch t.

$$\mathcal{L}_{\mathcal{BP}} = \beta(t) \cdot \mathcal{L}_{\mathcal{BT}} + (1 - \beta(t)) \cdot \mathcal{L}_{\mathcal{NT}} \tag{4}$$

The weighting function $\beta(t)$ is inspired by the Profile Weight Functions (Ravi et al., 2022) and is designed to optimize a multi-task learning problem - a combination of two tasks (one self-supervised and one fully supervised) - while avoiding training convergence issues. In particular, this function heavily weights the loss associated with the first task at the beginning of training and gradually increases the focus entirely on the second task towards the end. In our pipeline, this is achieved using a linear function ($\beta(t) = 1 - \frac{t}{H}$ with H the total number of epochs) which leads to a two-phase training process: the first phase focuses on making the framework invariant to different variability of the scans, while the latter phase focuses on learning contrastive features to distinguish scans of different participants.

2.3. Image transformations

In this session, we describe the transformations used in our framework. These transformations (implemented using MONAI (Cardoso et al., 2022)) augment our datasets and generate distorted versions of scans. They are divided into two categories: i) intensity-based –aimed to tackle domain shift variability caused by different contrasts or by different acquisition scanners, and ii) structural based –aimed to make the framework robust to variability related to the subject morphology (e.g., different subject rotations, ageing, disease progression). Our image distortions are obtained by a random sequence of these transformations. More specifically, we define a binary random vector $T = (\phi_1, \ldots, \phi_6)$ where each component ϕ_i specifies whether or not the i-th transformation –listed in Table 1– is used. Formally, each component ϕ_i follows an independent Bernoulli distribution $\phi_i \sim \text{Bern}(p_i)$ with each p_i also specified in Table 1. We define $\mathcal{T} = \text{Bern}(p_1) \times \cdots \times \text{Bern}(p_n)$ as the distribution of the random vector T such that $T \sim \mathcal{T}$.

Transformation	Type	$\mathbf{p_i}$	Parameters
Negative of the image	Intensity-based	40%	-
Intensity shifts	Intensity-based	40%	sampled in $[-0.25, 0.25]$
Bias field	Intensity-based	30%	-
Rotations	Structural-based	100%	max 3°
Random black patches	Structural-based	40%	max 3 patch of 10×10 pixel
Elastic deformation	Structural-based	30%	magnitude range is $[1, 2]$

Table 1: Proposed transformations used for image distortion during training.

2.4. Default training settings

The encoder $\mathcal{R}$ is a ResNet-18 model pre-trained on ImageNet, with the classification layer removed. It generates an output of size $s = 512$. During training, we freeze the first two blocks of convolutional layers and fine-tune the remaining layers. The projection head P is a multilayer perceptron that maps the representations (of size $s = 512$) to an embedding space (of size $z = 2048$), followed by a batch normalization layer and a ReLU activation. After training, the projection head P is discarded and is not used to compute the image fingerprints. The mining procedure uses a semi-hard negative strategy that selects, in each triplet, the most similar negative sample that is less similar than the corresponding positive sample. To train our framework, we use the same configuration settings as (Zbontar et al., 2021). Specifically, we use a LARS optimizer with two different learning rates - one for the weights ($0.2 \cdot b/256$) and one for the biases ($0.0048 \cdot b/256$). The training begins with a linear warm-up of 10 epochs, then reduces the learning rate by a factor of 1000 using a cosine decay schedule. Training is performed for 180 epochs with a batch size of $b = 64$. The hyperparameters λ and τ are fixed at 0.0051 and 0.07, respectively. The validation set is used for early stopping. The code of our pipeline is available at https://github.com/DeepBrainPrint/DeepBrainPrint (anonymized for the review process).

3. Dataset

To validate our framework we used two different datasets. The first consists of 2D slices of skull-stripped T1-MRI scan from ADNI. Here, every subject has on average 10 ± 7 scans

distributed on an interval of 3 years. The second dataset is obtained synthetically from the first one, by applying different intensity-based transformations to the scans (negative, and random intensity-shift) which have the purpose to simulate the acquisition of different modalities/contrasts of the scans. We will refer to this dataset as SYNT-CONTR. To avoid testing the framework on easy scans that are close in time to the query image (i.e. repeated scans) we sample the datasets such that the follow-up scans of each participant have a distance of at least 180 days. After this data selection, both datasets have 795 scans each (588 used for training, 142 for testing, and 65 for validation).

4. Experimental results

Our experiments are designed to retrieve all the scans of a specific subject given a query image. Inspired by (Musgrave et al., 2020), we used the following protocol to evaluate the performance of our framework: for each scan, in each of the test sets, we retrieve the k most similar representation and we compute two main quality metrics: i) the mean average precision (mAP@R) which takes into account both the precision and the order of the top-k retrieved scans, and ii) the Recall@K (R@K), which indicates the percentage of testing examples whose top-K retrievals include at least one scan from the correct subject. The mathematical definitions of these metrics are reported in Appendix A. We compare our pipeline against state-of-the-art techniques on both the ADNI and SYNT-CONTR test sets. We present these results in Section 4.1 while in Section 4.2 we show some visual results of our pipeline. In Appendix B, we also report the result of our approach using different batch sizes (b=64 and b=128) and different formulas for the $\beta(t)$ function to demonstrate its contribution to the training process.

4.1. Comparison against state-of-the-art methods

We compare DeepBrainPrint against the following methods: i) a handcrafted solution that uses the SSIM index (Wang et al., 2004) as a distance metric between scans, two Self-Supervised ($\widehat{SS}$) learning techniques (BarlowTwins (Zbontar et al., 2021), SimCLR (Chen et al., 2020)), two standards Fully-Supervised ($\widehat{FS}$) machine learning techniques used for metric learning (NCA (Goldberger et al., 2004) and MLKR (Weinberger and Tesauro, 2007)), and three Fully-Supervised ($\widehat{FS}$) contrastive deep learning techniques (SoftTriple (Qian et al., 2019), ProxyNCA (Movshovitz-Attias et al., 2017), NT-Xent (Chen et al., 2020)) trained using the same backbone architecture employed in our supervised branch. Additionally, where possible, we decide to extend these approaches with our domain-specific Data Transformations ($\widehat{DT}$). The obtained results are reported in Table 2.

Self-supervised methods (BarlowTwins and SimCLR) perform poorly when used with their original image transformations (without $\widehat{DT}$ selected). However, their performance improves significantly when we replace their transformations with our proposed ones. On the ADNI dataset, Barlow Twins with our transformations increases the mAP@3 by 199%, and on the SYNT-CONTR dataset by 312%. Similarly, on the ADNI dataset, SimCLR with our transformations increases the mAP@3 by 176%, and on the SYNT-CONTR dataset by 162%. These results highlight the importance of selecting suitable image transformations for MRI scans, as well as the effectiveness of our proposed transformations.

	Settings			ADNI		SYNT-CONTR	
Method	$\widehat{\mathbf{FS}}$	$\widehat{\mathbf{SS}}$	$\widehat{\mathbf{DT}}$	**R@3**	**mAP@3**	**R@3**	**mAP@3**
SSIM-based (Wang et al., 2004)	No training			96.89	90.21	76.68	48.86
Barlow Twins (Zbontar et al., 2021)		✓		73.06	45.35	48.70	25.52
Barlow Twins with our transformations		✓	✓	97.41	90.47	92.23	79.62
SimCLR (Chen et al., 2020)		✓		68.39	38.47	51.30	24.55
SimCLR with our transformations		✓	✓	87.05	67.63	70.98	39.94
NCA (Goldberger et al., 2004)	✓		✓	96.89	90.34	72.02	48.10
MLKR (Weinberger and Tesauro, 2007)	✓		✓	96.37	90.03	72.02	48.07
SoftTriple (Qian et al., 2019)	✓		✓	98.45	91.97	96.89	87.64
Proxy-NCA (Movshovitz-Attias et al., 2017)	✓		✓	98.45	90.80	94.82	84.86
NT-Xent (Chen et al., 2020)	✓		✓	96.89	94.04	95.34	86.95
DeepBrainPrint (Proposed)	✓	✓	✓	**99.48**	**95.54**	**98.96**	**91.00**

Table 2: Performance (expressed as %) of our framework compared to other methods for the task of patient identification, when evaluated on two different datasets.

Surprisingly, the SSIM-based solution performs quite well on ADNI dataset (i.e., mAP@3 around 90%) However, its performance drops significantly (from 90% to 49% for mAP@3) when it is applied to SYNT-CONTR. This may be because this approach partially relies on comparing intensity values and fails to capture anatomical structure. NCA and MLKR exhibit similar behaviour, performing well on the ADNI but poorly on synthetic data. In this case, their failure may be due to their inability to learn complex structures, even if they are trained using our data transformation technique. We can conclude that SSIM, NCA, and MLKR work well on scans with the same contrast, but are not able to capture the anatomical details of brain structures when applied to images with different contrast.

The fully-supervised contrastive deep learning methods (SoftTriple, ProxyNCA, NT-Xent) seem to overcome this limitation. They show very high performance on both ADNI (mAP@3 equal to 92%, 91%, and 94%, respectively) and synthetic data (mAP@3 equal to 88%, 85%, and 87%, respectively). Finally, the best performance is achieved by our hybrid method (mAP@3= 96% on ADNI and 91% on SYNT-CONTR). In comparison to the best supervised methods (NT-Xent on ADNI and SoftTriple on SYNT-CONTR), our solution improves mAP@3 by +1.5 and +3.4 percentage points, respectively. In comparison to the best self-supervised method (Barlow Twins with our transformations on both datasets), our solution improves mAP@3 by +5.0 and +11.3 percentage points. This demonstrates the superiority of our approach, which appears to capture the benefits of both self-supervised and supervised techniques. The statistical significance of the obtained improvements was assessed using a paired t-test, with all p-values less than 0.0001.

4.2. Visual results

In Figure 2 we show examples of outcomes from our pipeline obtained using six different test images from the ADNI dataset. On the left, we selected three queries where the retrieved scans were all correct. On the right, instead, we selected three queries having a few incorrect results (highlighted in red). As we can see in the latter case, although the retrieved scans belong to subjects with the wrong ID, their brain structures are very

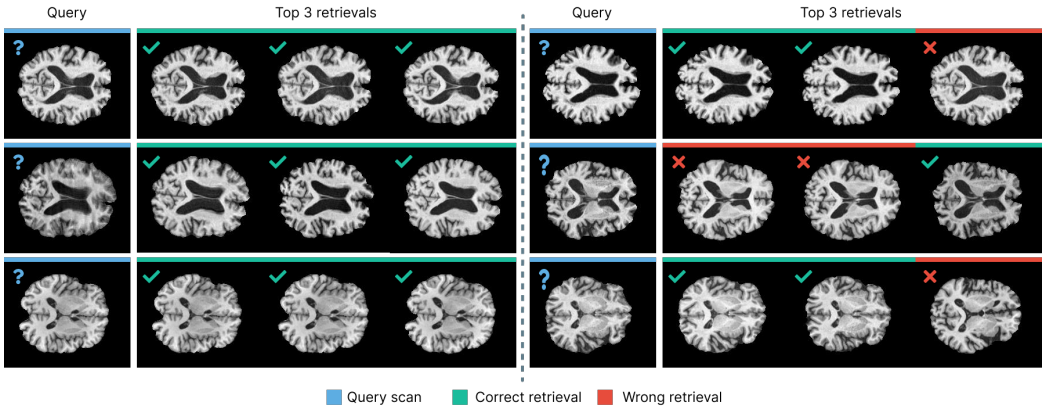

Figure 2: Example of outcomes from our pipeline on six test data extracted from ADNI.

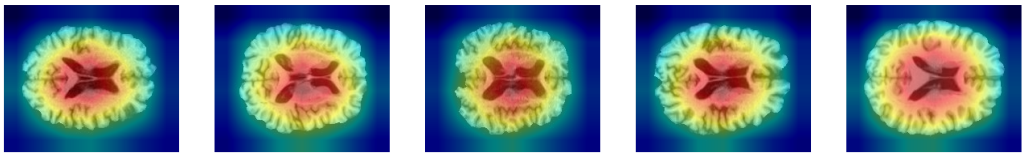

Figure 3: Examples of saliency maps showing the brain region that contributes most to the retrieval task.

similar to the query image. This indicates that our approach can be used to retrieve subjects having similar anatomical details. Finally, in Figure 3 we display the result of an explainable AI framework called Grad-CAM (Selvaraju et al., 2017) which creates a saliency map to visualise which regions of the brain contribute more to the retrieval task. This also indicates which structures are more unique among all the subjects. Interestingly, our model seems to focus more on the ventricles and less on the fine detail of the cortex. For more detail on Grad-CAM, please refer to Appendix C.

5. Conclusion and future work

In this work, we present DeepBrainPrint, a novel semi-self-supervised pipeline for creating fingerprints of brain MRI scans in real-time and efficiently retrieving images of the same subject within a large dataset. Our results demonstrate that DeepBrainPrint can accurately retrieve previous scans from the same subject, even when the images have different contrast. This demonstrates DeepBrainPrint's ability to capture the similarity of anatomical details of brain structures directly from MRI.

We believe that our pipeline can be useful for a range of different applications that require the retrieval of scans based on their content, including searching for similar scans that may not be from the same subject, (i.e. scans with similar brain shapes, lesions, or atrophy). This can be useful for exploiting image-based findings obtained on scans from subjects with similar anatomical details, to support diagnostic decisions or suggest treatments that have been effective for a similar disease subtype/stage.

In future work, we plan to extend DeepBrainPrint to use multiple slices or the entire 3D volume to further improve model performance.

References

Hanaa Mansour Al-bory and Mohammed M Elsheh. Development of medical image retrieval model based on minimum cross entropy thresholding. *Development*, 7(3):108–115, 2018.

M Jorge Cardoso, Wenqi Li, Richard Brown, Nic Ma, Eric Kerfoot, Yiheng Wang, Benjamin Murrey, Andriy Myronenko, Can Zhao, Dong Yang, et al. Monai: An open-source framework for deep learning in healthcare. *arXiv preprint arXiv:2211.02701*, 2022.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, pages 1597–1607. PMLR, 2020.

Sumit Chopra, Raia Hadsell, and Yann LeCun. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546. IEEE, 2005.

Andreia V Faria, Kenichi Oishi, Shoko Yoshida, Argye Hillis, Michael I Miller, and Susumu Mori. Content-based image retrieval for brain mri: An image-searching engine and population-based analysis to utilize past clinical data for future diagnosis. *NeuroImage: Clinical*, 7:367–376, 2015.

Jacob Goldberger, Geoffrey E Hinton, Sam Roweis, and Russ R Salakhutdinov. Neighbourhood components analysis. In L. Saul, Y. Weiss, and L. Bottou, editors, *Advances in Neural Information Processing Systems*, volume 17. MIT Press, 2004.

Elad Hoffer and Nir Ailon. Deep metric learning using triplet network. In *International Workshop on Similarity-based Pattern Recognition*, pages 84–92. Springer, 2015.

Vijay Kumar BG, Gustavo Carneiro, and Ian Reid. Learning local image descriptors with deep siamese and triplet convolutional networks by minimising global loss functions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Tecognition*, pages 5385–5394, 2016.

Yair Movshovitz-Attias, Alexander Toshev, Thomas K Leung, Sergey Ioffe, and Saurabh Singh. No fuss distance metric learning using proxies. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 360–368, 2017.

Kevin Musgrave, Serge Belongie, and Ser-Nam Lim. A metric learning reality check. In *European Conference on Computer Vision*, pages 681–699. Springer, 2020.

Qi Qian, Lei Shang, Baigui Sun, Juhua Hu, Hao Li, and Rong Jin. Softtriple loss: Deep metric learning without triplet sampling. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 6450–6458, 2019.

Daniele Ravi, Stefano B Blumberg, Silvia Ingala, Frederik Barkhof, Daniel C Alexander, Neil P Oxtoby, Alzheimer's Disease Neuroimaging Initiative, et al. Degenerative adversarial neuroimage nets for brain scan simulations: Application in ageing and dementia. *Medical Image Analysis*, 75:102257, 2022.

Christopher G Schwarz, Walter K Kremers, Terry M Therneau, Richard R Sharp, Jeffrey L Gunter, Prashanthi Vemuri, Arvin Arani, Anthony J Spychalla, Kejal Kantarci, David S Knopman, et al. Identification of anonymous mri research participants with face-recognition software. *New England Journal of Medicine*, 381(17):1684–1686, 2019.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-cam: Visual explanations from deep networks via gradient-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 618–626, 2017.

Christian Wachinger, Polina Golland, William Kremen, Bruce Fischl, Martin Reuter, Alzheimer's Disease Neuroimaging Initiative, et al. Brainprint: A discriminative characterization of brain morphology. *NeuroImage*, 109:232–248, 2015.

Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004.

Kilian Q. Weinberger and Gerald Tesauro. Metric learning for kernel regression. In Marina Meila and Xiaotong Shen, editors, *Proceedings of the Eleventh International Conference on Artificial Intelligence and Statistics*, volume 2 of *Proceedings of Machine Learning Research*, pages 612–619, San Juan, Puerto Rico, 21–24 Mar 2007. PMLR.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stéphane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *International Conference on Machine Learning*, pages 12310–12320. PMLR, 2021.

Appendix A. Evaluation metrics

In this section, we describe the quality metrics used in Section 4 to evaluate our pipeline. Let D be the test set. We denote with $x_q^{(i)}$ the i-th scan of the test set, and with $x_1^{(i)}, \ldots, x_k^{(i)}$ its k-most similar scans retrieved from $D \setminus \{x_q^{(i)}\}$. Also, $y_j^{(i)}$ will represent the patient ID of the scan $x_j^{(i)}$. Let $\mathbb{1}[\cdot]$ be a function that returns 1 if the inner condition is satisfied, and 0 otherwise. The mean average precision (mAP@R) –which is the first quality metric used– is computed as follows:

$$\text{mAP@R}(D) = \frac{1}{|D|} \sum_{i=1}^{|D|} \text{AP@R}(x_q^{(i)}, x_1^{(i)}, \ldots, x_R^{(i)}); \tag{5}$$

where AP@R represents the average precision score:

$$\text{AP@R}(x_q^{(i)}, x_1^{(i)}, \ldots, x_R^{(i)}) = \frac{\sum_{i=1}^{R} \mathbb{1}[y_q^{(i)} = y_i^{(i)}] \left[\frac{1}{i} \sum_{j=1}^{i} \mathbb{1}[y_q^{(i)} = y_j^{(i)}]\right]}{R}. \tag{6}$$

The mathematical definition of our second quality metric R@K (Recall@K) is instead:

$$\text{R@k}(D) = \frac{1}{|D|} \sum_{i=1}^{|D|} \mathbb{1}\left[y_q^{(i)} \in \{y_1^{(i)}, \ldots, y_k^{(i)}\}\right]. \tag{7}$$

Appendix B. Pipeline setup

B.1. Experiment with different weighting functions

In this section, we provide the performances obtained by our pipeline using four different formulations of $\beta(t)$. The first formulation weights the two terms $\mathcal{L}_{\mathcal{NT}}, \mathcal{L}_{\mathcal{BT}}$ of our total loss using a constant function (i.e. $\beta(t) = 0.5$). The second formulation optimizes one term of the loss at a time by using the following step function:

$$\beta(t) = \begin{cases} 1 & \text{if } t < \delta \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

The third formulation is an extension of the previous step function, obtained by iteratively optimising one term at a time on intervals of Δt epochs:

$$\beta(t) = \begin{cases} 1 & \text{if } \lfloor t/\Delta t \rfloor \text{ is even} \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

The last formulation weights the terms of the loss using a linear function over training defined as follows:

$$\beta(t) = 1 - \frac{t}{H} \tag{10}$$

where H is the total number of epochs. The hyperparameters of these functions (i.e. Δ and δ) are optimized using a grid search technique performed on the validation set. The results, reported in Table 3, indicate that the best results are obtained when the linear function is used.

		ADNI		**SYNT-CONTR**	
Weighting function	**Hyperparameter**	**R@3**	**mAP@3**	**R@3**	**mAP@3**
Constant	-	96.89	90.82	89.64	75.39
Step	$\delta = 30$	96.89	88.89	90.67	74.38
Iterative Step	$\Delta t = 30$	96.37	89.75	89.64	78.09
Linear	-	**99.48**	**95.54**	**98.96**	**91.00**

Table 3: Performance of our framework (expressed as %) by using different weighting functions.

B.2. Experiment with different batch size

We trained our pipeline using two different batch sizes $b = 64$ and $b = 128$ and we found that $b = 64$ provides the best results. These performances are reported in Table 4.

	ADNI		SYNT-CONTR	
Batch Size	**R@3**	**mAP@3**	**R@3**	**mAP@3**
128	98.45	93.51	97.93	90.31
64	**99.48**	**95.54**	**98.96**	**91.00**

Table 4: Performance of our framework (expressed as %) by using different batch size.

Appendix C. Explainability of DeepBrainPrint using Grad-CAM

To obtain the saliency map presented in Figure 3, we computed the Grad-CAM localization map (Selvaraju et al., 2017) for each element of the representation vector calculated from a specific input image and targeting the last convolutional layer of $\mathcal{R}$. We then averaged all of the obtained localization maps (some of which are shown in Figure 4) and normalized them in the range $[0, 1]$ to produce the final saliency map. Each point in this final map indicates the importance of the corresponding pixel of the input scan for the retrieval task.

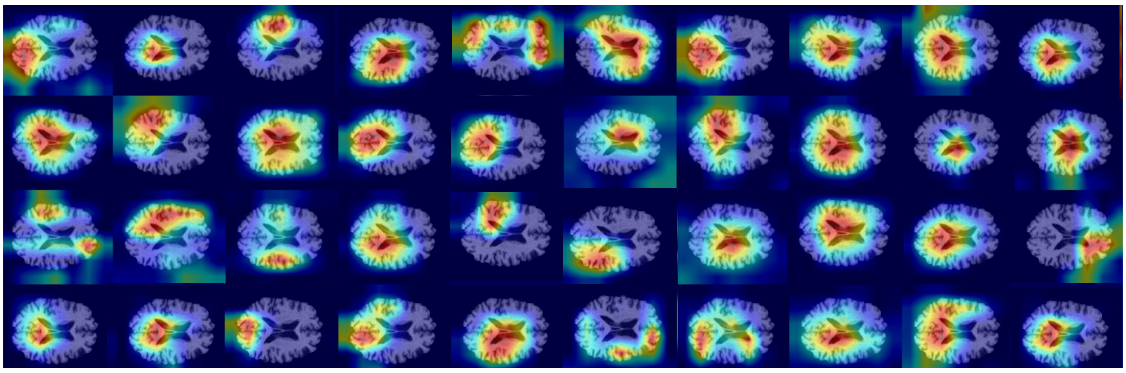

Figure 4: Examples of Grad-CAM activation maps computed on individual elements of the representation vector for a specific input image.