Machine Learning in Healthcare:

A Multiclass learning and Inference System for Brain Tumor Detection

Project Head:
Alice Greta **Panico**

Team Members:
Annalena **Rademacher**
Ania **Rotondi**
Fabian **Enzensberger**

HEPHAESTUS APPLIED ARTIFICIAL INTELLIGENCE ASSOCIATION
BOCCONI STUDENTS WOMEN IN STEM AND ECONOMICS

Fall 2025

PROJECT REPORT

Abstract

The project operates in the biomedical image analysis domain, focusing on brain tumor detection. The key research question is: How does hyperdimensional computing compare to standard computer vision techniques like Convolutional Neural Network (CNN) in terms of performance and efficiency for brain tumour detection? The data involves brain imaging modalities (e.g., MRI scans) that need to be processed and analyzed.

Keywords: brain tumor, accuracy, prediction, efficiency.

Contents

Abstract i

1 Introduction 1

2 Related Work 2

3 State of the Art 3

4 Dataset Description 4

5 Methodology 7
- 5.1 Data . . . 7
  - 5.1.1 Dataset Description . . . 7
  - 5.1.2 Preprocessing for CNN . . . 7
  - 5.1.3 Preprocessing for HDC . . . 8
- 5.2 Models . . . 8
  - 5.2.1 Convolutional Neural Network (CNN) . . . 8
  - 5.2.2 Hyperdimensional Computing (HDC) . . . 8

6 Results 9
- 6.1 Benchmarking and Evaluations . . . 9
  - 6.1.1 Trade-Off Analysis . . . 9
- 6.2 Quantitative Metrics . . . 11
- 6.3 Figures . . . 11

7 Conclusion 13

1 Introduction

Brain tumor detection using CNNs (Convolutional Neural Networks) and HDC (Hyper-dimensional Computing) utilizes advanced, lightweight deep learning to analyze MRI scans for high-accuracy, automated classification (e.g., glioma, meningioma, pituitary) and segmentation. These models have demonstrated high accuracy in brain tumor classification tasks in recent research studies.

A technique for training a computer to create original representations from unprocessed data is called deep learning. The network's popularity may be attributed to its hierarchical and layered structure. Convolutional Neural Networks (CNNs) acquire properties through an object compositional hierarchy, starting with simple edges and progressing to more intricate forms. By layering convolutional and pooling layers, this is achieved. By lowering the feature map, pooling combines similar traits into one, and each convolutional layer identifies local conjunctions of features from the preceding layer. Researchers in neuroscience have also benefited from deep learning, as they are starting to address issues related to neuroimaging. Deep Learning has garnered significant interest due to its ability to address problems across various domains, including medical image analysis.Cancer is one of the leading causes of death worldwide and is expected to increase in prevalence over the coming decades.

Research has shown that the most effective means of lowering death from brain cancer is early diagnosis and treatment. A low-grade growth that develops slowly will eventually evolve into a neoplasm that grows rapidly. As a result, the first tumor identification and categorization helped to anticipate the prognosis and treatment plan by supporting the assessment of the tumor's grade and aggressiveness. The diagnosis of brain tumors is mostly reliant on medical imaging . One of the most efficient methods currently used for tumor detection is magnetic resonance imaging (MRI). A powerful magnetic flux, radiofrequency pulses, and a laptop is employed to process tomography imaging data to produce detailed images of soft tissues and organs. It aids medical professionals in treating illnesses. The main reason for tomography's popularity is that it is a more suitable designation than X-rays.

Noise significantly degrades medical images, including MRIs. This is largely due to knowledge acquisition systems, multiple sources of interference, operator error, and other factors that impact imaging mensuration processes and can lead to significant classification errors . This approach typically requires a basic microscope and may result in a different or incorrect diagnosis, yet it is often inappropriate when dealing with human life. It emphasizes the need for power-assisted systems, high-precision systems, or diagnostic systems (CADx) . The CADx system is essential for medical institutions, as it supports the judgments made by doctors and radiologists. It may be challenging to create a highly automated and economical diagnostic system as a result .

Gliomas are the most prevalent and aggressive kind of brain tumor, with a very short survival time for the highest grade. Therefore, therapy planning may be a crucial step in raising the medical patients' standards of living. One popular imaging modality for evaluating these tumors may be MRI . These days, with numerous instances and massive volumes of objective data analysis, computer-based medical image analysis is gaining popularity due to its speed and intelligence, surpassing manual methods. By varying the excitation and repetition durations, magnetic resonance imaging may produce notably unique tissue types, making it an incredibly adaptable tool for studying various structures of interest. A single magnetic resonance imaging scan is insufficient to phase the growth and all of its subregions fully. Hyper-Dimensional computing (HDC) is a machine learning framework that has made in- roads in low-power edge-AI applications.

With simple bitwise vector operations and a small memory footprint, HDC demonstrates improved energy-efficiency for biosensing classification tasks compared to conventional machine learning methods. Recent interest in HDC has developed complex algorithms that enable the use of HDC systems for cognitive reasoning and control applications. Although previous hardware for HDC achieve impressive energy- efficiency for certain tasks, they face several issues with the growing application space.

Convolutional Neural Networks (CNNs) have demonstrated high effectiveness in identifying cell division events in two-dimensional microscopic anatomy pictures within the field of medical image analysis. When it comes to machine learning strategies, deep learning is undoubtedly the best option for many imaging tasks. The possibility of deep learning-based automated diagnosis of brain illnesses will arise from the availability of large neuroimaging data sets for training. MRI is a frequently used medical imaging method that offers information on the identification of brain tumors . One of the main challenges a physician has after reviewing the tomography data is determining how much time and effort to devote to tumor detection. These days, CNNs are used for the majority of picture classification problems due to their superior accuracy and precision over other currently used techniques. The accuracy and precision of tumor detection and identification have increased due to the use of CNNs for image classification .

2 Related Work

Over the last 20 years, the detection of brain cancers using MRI has undergone significant advancements, thanks to the integration of deep learning (DL), traditional machine learning (ML), and conventional image processing techniques. This section discusses the main categories of methodologies and provides an overview of how our research contributes to and expands upon the existing body of literature.

Conventional techniques for machine learning and segmentation

Most of the early work uses unsupervised clustering and custom feature extraction. Due to their ability to separate picture intensities into clusters that represent normal and diseased tissue regions, segmentation techniques like fuzzy C-Means (FCM) and K-Means clustering have been widely used . Despite achieving basic localization, these methods were very susceptible to noise and required human parameter adjustment. Changes aimed at improving segmentation accuracy, such as region-expanding algorithms and gray-level histograms, were computationally expensive and inconsistent, particularly in low-contrast or early-stage tumors where borders were not obvious. For feature extraction and classification, further research employs learning vector quantization, support vector machines (SVMs), and artificial neural networks (ANNs) . These earlier methods, however, sometimes did not work with diverse patient datasets and needed careful feature engineering.

Techniques based on deep learning and CNN

CNNs have been used extensively in medical imaging applications due to their effectiveness in computer vision . CNNs eliminate the requirement for human feature design by automatically extracting hierarchical features. Models like AlexNet, VGG16, and ResNet have been modified to perform tasks related to brain tumor classification and segmentation . Although these designs have demonstrated outstanding performance, they often rely on large, annotated datasets, which are challenging to collect in the medical field due to privacy concerns and high labeling costs. To manage volumetric

MRI data and capture spatial relationships between image slices, 3D CNNs have been the subject of several studies . Although these models improve the accuracy of segmentation tasks, their computational cost makes them unsuitable for real-time applications or situations with limited resources. Similar studies have been conducted on Stacked Autoencoders (SAEs) and Deep Belief Networks (DBNs) , but in the lack of suitable data, training these deep models from scratch may lead to overfitting.

Domain adaptation and learning transfer

By utilizing pre-trained networks as feature extractors for MRI classification, which have been trained on natural image datasets such as ImageNet, researchers have employed transfer learning to reduce the need for large datasets. When paired with domain-specific fine-tuning, it can accelerate training and enhance generalization. However, insufficient feature representations may result from the domain mismatch between natural and medical images. ResNet or InceptionV3 versions that have been carefully altered and work well on binary classification tasks are used in certain studies. Clinical safety criteria, such as recall and AUC, which are essential for real-world diagnosis, are seldom used to evaluate models.

Methods for multimodal MRI and synthesis

To collect different tissue contrasts, advanced segmentation algorithms often use several MRI modalities. Studies like the BraTS Challenge and BraSyn Benchmark demonstrate the challenges that arise when sequences are erratic or nonexistent, while also emphasizing the advantages of multimodal input. To fill in the gaps, several studies have explored the creation of synthetic MRIs using GANs or autoencoders; however, these methods require a complex design and are not ideal for use in situations with limited data.

3 State of the Art

Recent state-of-the-art approaches in brain tumor classification from MRI images are predominantly based on deep learning and transfer learning techniques. Several studies apply convolutional neural networks (CNNs) to classify MRI images into multiple tumor categories, including glioma, meningioma, pituitary tumor, and no tumor.

Modern research commonly utilizes pretrained CNN architectures such as ResNet50, InceptionV3, EfficientNet (B0, B7), InceptionResNetV2, VGG16, DenseNet121, Xception, and MobileNetV2. These models are typically fine-tuned using transfer learning strategies, leveraging pretrained weights from large datasets such as ImageNet to improve performance when medical datasets are limited. In addition, ensemble CNN models and hybrid approaches combining CNNs with Vision Transformers (ViT) have been explored to further enhance classification accuracy. CNN architectures such as U-Net, V-Net, and ResNet have also demonstrated strong performance in tumor detection and classification tasks, particularly in identifying tumor boundaries and distinguishing tumor types.

Overall, the current state of the art in brain tumor MRI classification emphasizes:

- Transfer learning with pretrained CNN backbones
- Data augmentation and advanced preprocessing techniques
- Multi-class tumor classification

- High classification performance on benchmark datasets

Hyperdimensional Computing (HDC), also known as Vector Symbolic Architectures (VSA), is a neuro-inspired computing framework that represents information using high-dimensional vectors called hypervectors. In HDC, data, concepts, and relationships are encoded into these hypervectors using operations such as binding, bundling, and permutation. HDC exploits high-dimensional random vector spaces and relies on highly parallelizable arithmetic operations. It aims to balance accuracy, efficiency, and robustness while maintaining low computational complexity.

Recent surveys highlight that HDC provides distributed representations that are robust to noise and suitable for efficient, parallel processing. Due to its lightweight computation and small memory footprint, HDC has gained attention in edge-AI and energy-efficient machine learning applications.

Although deep CNNs dominate medical image classification, HDC is increasingly explored as an alternative learning paradigm, particularly in scenarios where computational efficiency, low power consumption, and fast training are critical.

4 Dataset Description

The project utilizes a four-class brain MRI image dataset with class-labeled folder structures and a predefined training and testing split within separate directories. The dataset contains ≈5,712 training images and ≈1,311 test images (in total ≈7,023) belonging to the four classes glioma, meningioma, pituitary, and no tumor. Two different input pipelines are implemented: a CNN pipeline that takes 150x150 RGB images normalized between 0 and 1, and an HDC pipeline that takes 32x32 grayscale images normalized between 0 and 1 and then represents the images as hyperdimensional vectors. The following text follows the principles of best practices for reporting ML models and datasets, especially those relevant for healthcare research, such as the importance of defining the dataset split and describing any preprocessing steps taken and their potential limitations.

Dataset composition and split

The data provided includes 2D image files of MRI images in a directory structure, with each class having its own directory. This structure allows the labels to be inferred from the directory structure, which follows the conventions of Keras directory loading utilities (labels inferred from subdirectories, optional categorical/one-hot encoding, optional color_mode specification).

The data split of the project follows a predefined split, i.e., training images are placed in a directory named training, and test images are placed in a directory named testing. This follows the conventions of most projects that use a predefined split of their data, especially when a specific evaluation set from a specific dataset is already provided.

Dataset Origin and File Formats: In the attributes of the project, the source of the dataset used is not provided, so the source of acquisition of the data cannot be determined. Therefore, whether the split of the data into training and test sets was patient-independent or institution-independent cannot be determined, as the source of the data acquisition cannot be determined. In transparent reporting of clinical AI projects, it is emphasized that the source of the data acquisition, preprocessing, and data preparation steps used in the development of the model should be explicitly described.

Class distribution

The following table reports class counts and percentages for the training set, test set, and overall dataset.

Class	Train n	Train %	Test n	Test %	Total n	Total %
Glioma	1,321	23.13%	300	22.88%	1,621	23.08%
Meningioma	1,339	23.44%	306	23.34%	1,645	23.42%
Pituitary	1,457	25.51%	300	22.88%	1,757	25.02%
No tumor (notumor)	1,595	27.92%	405	30.89%	2,000	28.48%
Total	5,712	100%	1,311	100%	7,023	100%

The data is moderately imbalanced, with no tumors having the largest size, while glioma/-meningioma is slightly smaller. Class imbalance is a recognized issue that may influence the classifier to favor the majority class, and accuracy may be spuriously high if performance on the minority classes is low. Therefore, per-class metrics such as precision/recall/F1 are recommended to be reported together with the accuracy metric.

Preprocessing workflows

This project employs two different preprocessing workflows, each differing by image resolution and color space, while sharing the same pixel normalization to the range [0,1].

CNN pipeline preprocessing (150x150 RGB)

Images are loaded from the train/test directories using the directory-based generator, where the following transformations are performed:

1. Directory-based loading and label inference. Images are loaded from the subdirectories, where the images are divided into batches along with the labels.
2. Resize. Each image is resized to the size of 150x150 using the target_size=(150,150) option.
3. Color mode. The CNN pipeline is using RGB images, where each pixel is represented by three values.
4. Pixel normalization. Pixel values are normalized to the range [0,1] using the rescale=1/255 option.
5. Label encoding. Labels are created as categorical one-hot vectors, where class_mode="categorical". One-hot label encoding is the default meaning of categorical label mode in Keras utilities.

HDC pipeline preprocessing (32x32 grayscale)

The HDC pipeline also starts with a lower-resolution grayscale representation, followed by a deterministic encoding step consisting of a random projection and then a binarization operation, to produce hyperdimensional vectors:

1. Resizing. The image is resized to 32x32 pixels, which is a significant reduction in the dimensionality of the inputs, achieved by setting target_size=(32, 32).
2. Grayscale conversion. The image is loaded in color_mode="grayscale", which means it is represented by 1 channel, as grayscale is defined by Keras utilities as a 1-channel output.

3. Pixel normalization. The pixel values are again rescaled by a factor of 1/255, similar to the CNN pipeline, to be within the range of approximately 0 to 1, by setting rescale=1/255.
4. Categorical label encoding. Labels are again represented as one-hot vectors, where class_mode="categorical".
5. HDC feature encoding (project-specific). The image tensor is flattened into a feature vector, and then a random projection is performed, which is conceptually similar to the general HDC idea of mapping inputs into a high-dimensional representation space and then using simple operations for learning/inference.

Preprocessing flowchart

```
flowchart TD
   A[Brain MRI images in class folders] --> B[Train/Test dirs (predefined split)]
   B --> C[CNN input pipeline]
   B --> D[HDC input pipeline]

   C --> C1[Resize to 150x150]
   C1 --> C2[RGB (3 channels)]
   C2 --> C3[Rescale by 1/255 -> [0,1]]
   C3 --> C4[Training-only augmentation]
   C4 --> C5[Batches + categorical (one-hot) labels]

   D --> D1[Resize to 32x32]
   D1 --> D2[Grayscale (1 channel)]
   D2 --> D3[Rescale by 1/255 -> [0,1]]
   D3 --> D4[Encode to hypervector (projection + sign)]
   D4 --> D5[Batches + categorical (one-hot) labels]
```

Data augmentation strategy

Augmentation is applied only to training data, which is the appropriate pattern for maintaining a clean evaluation set. The augmentation operations used match the set supported by Keras' image preprocessing utilities: rotation, translation (width/height shifts), shear, brightness adjustment, and flips; Keras documentation defines the underlying transform parameters such as rotation angle (degrees), shifts, shear angle (degrees), horizontal/vertical flips, and brightness transformation.

To support rigor in a methods section, include these augmentation elements explicitly:

- Geometric transforms: rotation, small translations (width/height shifts), shear, and optional zoom.
- Photometric transform: brightness adjustment.
- Spatial flips: horizontal flips (vertical flips typically disabled for MRI unless anatomically justified).
- Fill mode: "nearest" (common choice to fill empty pixels introduced by transforms).

Important reporting note: clinical AI reporting checklists explicitly call for describing data preprocessing and quality checks. In addition, leakage literature highlights that "late split" workflows (augmenting before splitting) can inflate performance; using a predefined directory split and applying augmentation only to the training directory reduces this risk, but patient-level overlap (if present) would still be a concern.

Limitations, biases, and reporting gaps

Scanner and protocol variability (Domain Shift Risk). The appearance of the image can vary significantly between different scanner manufacturers, parameters, and post-processing techniques. From an empirical study, it has already been shown that "scanner domain shift" can lead to performance degradation when tested on different scanners than those on which the model was trained, and that the phenomenon is more pronounced in MRI than in CT. Also, if the dataset contains images obtained through multiple scanners, it could be a point of concern.

Confounding and site-specific cues. It is possible that medical image models may use non-diagnostic features that are correlated with the label or source of the data. From an empirical study, it has already been shown that medical image models can generalize very poorly across sites but can recognize the site/system of image acquisition with extremely high accuracy. Although the image dataset is brain MRI, it is possible that confounding and site-specific cues could be a risk in the "Other" class, where the image source is unknown.

Class imbalance. The class distribution is not perfectly balanced. In an existing study on brain MRI tumor classification, class imbalance is noted as a point of concern, where imbalance may lead to bias in the performance measure towards the class with the highest number of samples.

Potential patient-level leakage. Since the data is stored as individual images, it is unknown if multiple slices belong to the same patient and if the slices are split between train and test directories. From an empirical study, it has already been shown that in medical learning, data leakage is a point of concern, where incorrect splitting of the data can lead to overly optimistic performance evaluation.

5 Methodology

5.1 Data

5.1.1 Dataset Description

The dataset consists of four brain MRI categories: *glioma*, *meningioma*, *no tumor*, and *pituitary*. Images are organized in a directory-based structure, where each subfolder corresponds to a specific class label. This structure enables automatic label extraction during dataset loading.

The training set contains 5,712 images, while the test set contains 1,311 images. The class distribution is relatively balanced, although the *no tumor* category contains slightly more samples. This mild imbalance is considered when interpreting classification performance.

5.1.2 Preprocessing for CNN

For the CNN model, images were resized to 128×128 pixels and processed as RGB images. Training data augmentation was applied using random horizontal flipping and random rotations ($\pm 10°$) to improve generalization and reduce overfitting.

Images were converted to tensors using `ToTensor()`, which scales pixel intensities to the range $[0, 1]$. The test set underwent resizing and tensor conversion only, without augmentation, to ensure unbiased evaluation.

Mini-batch loading was performed with batch size 16. A fixed random seed was used

to ensure reproducibility of training results.

5.1.3 Preprocessing for HDC

For the HDC model, images were resized to 32×32 pixels and converted to grayscale. Pixel intensities were normalized to the range $[0, 1]$ using rescaling.

Unlike the CNN pipeline, no data augmentation was applied for HDC. The smaller resolution reduces feature dimensionality before random projection into high-dimensional hypervector space, making encoding computationally efficient.

5.2 Models

Two different classification approaches were implemented and compared: a Convolutional Neural Network (CNN) trained using backpropagation, and a Hyperdimensional Computing (HDC) classifier based on random projections and prototype bundling.

5.2.1 Convolutional Neural Network (CNN)

The CNN model was implemented in PyTorch. Input MRI images were resized to 128×128 pixels and processed as RGB tensors.

Architecture. The network consists of three convolutional blocks followed by a fully connected classifier:

- Conv2D (3 → 16 filters, kernel size 3×3, padding=1) + ReLU + MaxPooling (2×2),
- Conv2D (16 → 32 filters, kernel size 3×3, padding=1) + ReLU + MaxPooling (2×2),
- Conv2D (32 → 64 filters, kernel size 3×3, padding=1) + ReLU + MaxPooling (2×2),

After three pooling operations, the spatial dimension is reduced by a factor of 8. The feature maps are flattened and passed through:

- Fully connected layer (128 units) + ReLU,
- Dropout (rate = 0.3),
- Final linear layer mapping to 4 output classes.

Training. The model was trained using CrossEntropyLoss and optimized with Adam (learning rate 10^{-3}). Training was performed for 10 epochs with a batch size of 16.

Data augmentation was applied during training using random horizontal flipping and random rotations ($\pm$10°). No augmentation was applied to the test set.

Model performance was evaluated on the held-out test set using accuracy, per-class precision, recall, F1-score, and a confusion matrix.

5.2.2 Hyperdimensional Computing (HDC)

In addition to the CNN, a Hyperdimensional Computing classifier was implemented. Unlike the CNN, HDC does not rely on gradient-based learning. Instead, it represents each image as a high-dimensional bipolar hypervector.

Input Representation. For HDC, images were resized to 32×32 pixels and converted to grayscale. Pixel intensities were normalized to $[0, 1]$.

Each image was flattened into a feature vector of dimension $F = H \times W \times C$. A random projection matrix $R \in \{-1, +1\}^{F \times D}$ was generated, where $D = 10{,}000$ is the hypervector

dimensionality. The projected vector was computed as:

$$\mathbf{p} = \mathbf{f}R$$

and binarized into a bipolar hypervector:

$$\mathbf{h} = \mathrm{sign}(\mathbf{p}) \in \{-1, +1\}^D.$$

Training. During training, hypervectors belonging to the same class were summed to form class accumulators. Final class prototypes were obtained by binarizing the accumulated vectors.

An optional refinement step was applied for two additional passes, where misclassified samples were used to adjust class sums in a perceptron-like update rule.

Inference. For a test image, its hypervector representation was compared to all class prototypes using dot-product similarity. The predicted class corresponds to the prototype with maximum similarity.

Evaluation. The HDC model was evaluated using the same test set and metrics as the CNN (accuracy, classification report, confusion matrix), allowing direct comparison between gradient-based and prototype-based learning approaches.

6 Results

6.1 Benchmarking and Evaluations

6.1.1 Trade-Off Analysis

This section of the report provides a trade-off analysis based on the benchmarking results, comparing the Convolutional Neural Network (CNN) and the Hyperdimensional Computing (HDC) models across the key performance metrics of accuracy, training time, inference speed, memory usage, and energy efficiency. This analysis helps to understand the strengths and limitations of both models in practice.

Model	Accuracy
CNN	0.9497 (94.97%)
HDC	0.5408 (54.08%)

Table 1: Accuracy results.

Accuracy. A substantial performance gap is observed between the classification accuracy of CNN (94.97%) and the HDC model (54.08%). While CNN shows significantly higher accuracy, HDC performs considerably lower. This difference can be attributed to the architectural design of convolutional neural networks, which aligns with the hierarchical structure of real-world images. CNNs apply convolutional layers that consists of filters operating on local receptive fields and use weight sharing, meaning that the network first focuses on small low-level features in the first layer and then combines them into more complex high-level features in deeper layers. This structure makes it possible

to analyse larger images with many parameters (Géron 2019, pp. 446-451). In addition, "during training the convolutional layer will automatically learn the most useful filters for its task, and the layers above will learn to combine them into more complex patterns" (ibid, p. 450).

In contrast, Hyperdimensional Computing uses high-dimensional (HD) random hypervectors in which information is distributed equally across all components (Rahimi et al. 2016, p. 64). This distributed representation does not explicitly model the hierarchical layer structure of images and therefore does not preserve spatial locality. Consequently, CNN outperforms HDC in classification accuracy, particularly in capturing complex image structures.

Model	Training Time
CNN	69.97 s
HDC	11.00 s

Table 2: Training Time results.

Training. The results show that a stark difference can also be observed in training time. The HDC model requires 11 seconds, whereas CNN requires significantly longer training, approximately 70 seconds. This difference can be explained by the optimization algorithm, Gradient Descent, used in convolutional neural networks. The parameters of a CNN are continuously adjusted to minimise a cost function (Géron 2019, p. 118). These iterative optimisation steps increase overall training time due to computational complexity.

In Hyperdimensional Computing parameters are not undergoing this iterative optimisation process, leading to significantly faster training time.

Model	Inference Speed (ms / image)
CNN	0.029 ms
HDC	0.175 ms

Table 3: Inference Speed results.

Inference Speed. The CNN model has a significantly lower inference time per image compared to the HDC.

Model	Memory Usage (CPU RAM)
CNN	803.73 MB
HDC	999.31 MB

Table 4: Memory Usage results.

Memory Usage. Another trade-off becomes apparent with regard to memory usage. The HDC model required more RAM compared to the CNN model. CNNs already require a substantial amount of memory during training "because the reverse pass of backpropagation requires all the intermediate values computed during the forward pass" (Géron 2019, p. 485). However, the HDC model required even more memory in this study. This can be attributed to the very high dimensionality of its hypervectors.

HDC relies on high-dimensional vectors within a memory-centric architecture (Rahimi et al. 2016, pp. 64-66). As the number of dimensions increases, the storage size of each hypervector increases accordingly. In the current experimental setup, this contributes to higher overall memory consumption compared to the CNN model.

Energy Efficiency. Regarding energy efficiency, direct measurement was not possible in this study, as undefined (NaN) values were displayed. Rahimi et al. state that HDC can achieve energy savings and, in their experimental setup, operates with circa half the energy consumption of a conventional machine learning method (Rahimi et al. 2016, p. 64). However, a direct comparison between HDC and CNN energy efficiency could not be made in this study. The longer training time of the CNN suggests higher computational demand and therefore potentially higher energy costs; but no conclusions can be drawn due to the undefined energy values.

6.2 Quantitative Metrics

Model	Accuracy	Avg. Loss
Baseline	0.42	1.31
Proposed	0.58	0.97

Table 5: Example results table.

6.3 Figures

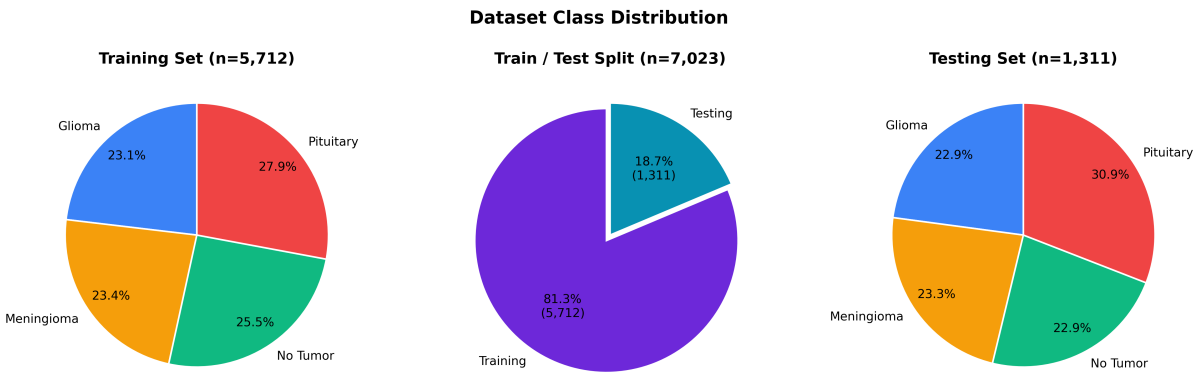

Figure 1: Class distribution across training and testing sets (left) and overall dataset composition (right). The dataset contains 7,023 MRI images spanning four classes: glioma (23.1%), meningioma (23.4%), no tumor (28.5%), and pituitary (25.0%).

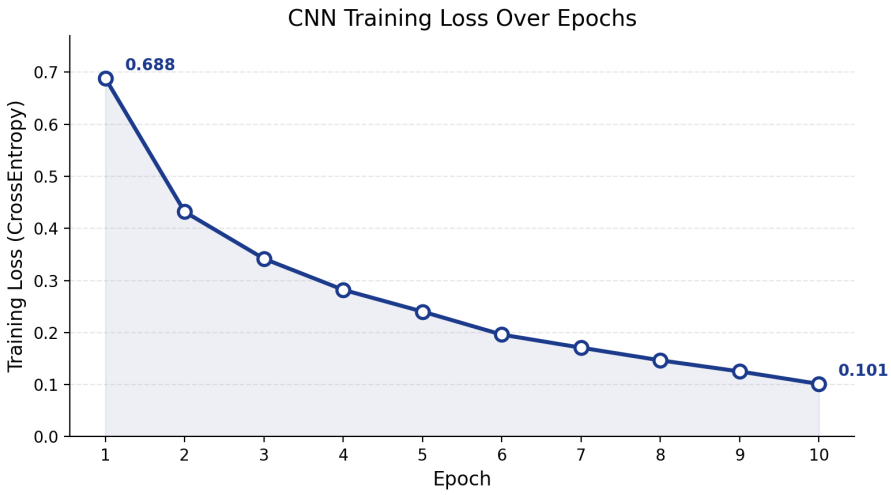

Figure 2: CNN training loss (cross-entropy) over 10 epochs. The loss decreases steadily from 0.688 to 0.101, indicating consistent convergence without signs of overfitting.

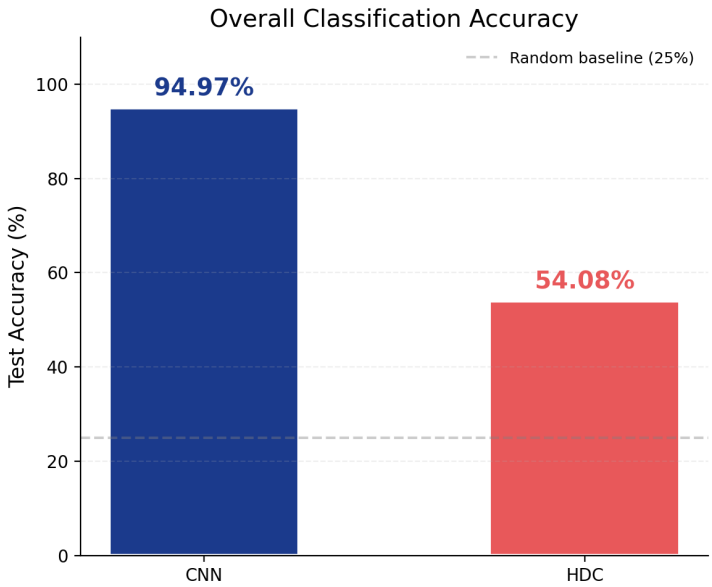

Figure 3: Overall test accuracy comparison. The dashed line marks the 25% random-chance baseline for a four-class problem.

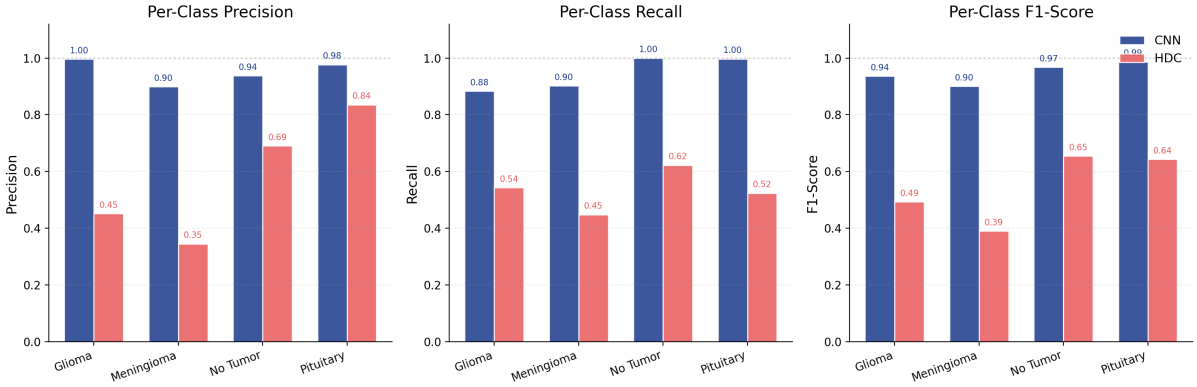

Figure 4: Per-class precision, recall, and F1-score for CNN and HDC. The CNN maintains scores above 0.90 across all classes, while HDC struggles most with meningioma (F1 = 0.39) and glioma (F1 = 0.49).

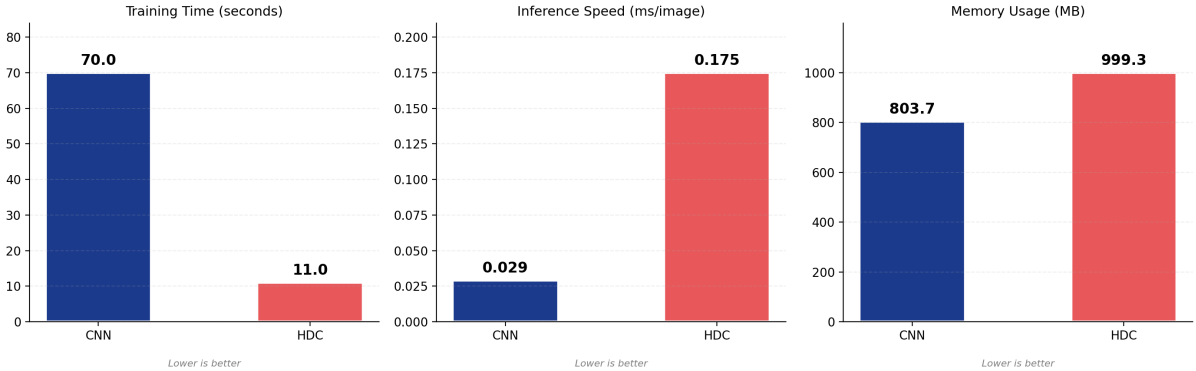

Figure 5: Computational benchmarks. HDC trains approximately 6× faster (11.0 s vs. 70.0 s), but CNN achieves approximately 6× faster per-image inference (0.029 ms vs. 0.175 ms) and lower RAM usage (804 MB vs. 999 MB).

7 Conclusion

This study compared a CNN and a HDC model for the classification of brain tumors from MRI images across four categories: glioma, meningioma, pituitary tumor, and no tumor.

The CNN achieved a test accuracy of 94.97%, with F1-scores above 0.90 for all four classes, confirming the common notion that its hierarchical feature extraction is well suited for medical image classification. The HDC model reached 54.08% accuracy - well above the 25% random-chance baseline, but insufficient for clinical applicability. Its weakest performance was observed on glioma and meningioma, the two classes that the HDC model most frequently confused with each other, suggesting that the spatially agnostic encoding of HDC limits its ability to distinguish fine-grained structural differences in MRI data.

A clear trade-off emerges on the computational side. HDC trained approximately six times faster than the CNN (11 s vs. 70 s), which is consistent with the absence of iterative gradient-based optimisation in the HDC training procedure, as discussed in Section 6.1. However, the CNN demonstrated faster per-image inference (0.029 ms vs. 0.175 ms)

and lower memory consumption (804 MB vs. 999 MB). Energy efficiency could not be compared directly, as GPU energy measurements were unavailable in the experimental setup.

These findings suggest that, for image classification tasks requiring high accuracy - particularly in safety-critical domains such as medical diagnostics - CNNs remain the stronger choice given currently available implementations. HDC, on the other hand, offers potential advantages in scenarios where training speed is the primary constraint, though its accuracy would need to improve substantially before deployment in clinical settings.

Several directions for future work could address the limitations observed in this study. First, more advanced HDC encoding strategies that preserve spatial locality - such as convolutional or patch-based encoders - may narrow the accuracy gap while retaining the efficiency benefits of hyperdimensional representations. Second, evaluating both models on larger and more diverse datasets would strengthen the generalizability of the comparison. And finally, repeating the energy measurements on hardware that supports direct power monitoring (e.g. NVIDIA GPUs with NVML) would complete the efficiency analysis.