University of Arkansas, Fayetteville

ScholarWorks@UARK

Computer Science and Computer Engineering Undergraduate Honors Theses

Computer Science and Computer Engineering

5-2021

Using Deep Learning to Analyze Materials in Medical Images

Carson Molder

Follow this and additional works at: https://scholarworks.uark.edu/csceuht

Part of the Artificial Intelligence and Robotics Commons, Bioimaging and Biomedical Optics Commons, Data Science Commons, and the Radiology Commons

Citation

Molder, C. (2021). Using Deep Learning to Analyze Materials in Medical Images. *Computer Science and Computer Engineering Undergraduate Honors Theses* Retrieved from https://scholarworks.uark.edu/csceuht/89

This Thesis is brought to you for free and open access by the Computer Science and Computer Engineering at ScholarWorks@UARK. It has been accepted for inclusion in Computer Science and Computer Engineering Undergraduate Honors Theses by an authorized administrator of ScholarWorks@UARK. For more information, please contact scholar@uark.edu.

Using Deep Learning to Analyze Materials in Medical Images

Using Deep Learning to Analyze Materials in Medical Images

An Undergraduate Honors College Thesis

in the

Department of Computer Science and Computer Engineering
College of Engineering
University of Arkansas
Fayetteville, AR

By

Carson Molder
cbmolder@uark.edu

April 12th 2021

Abstract

Modern deep learning architectures have become increasingly popular in medicine, especially for analyzing medical images. In some medical applications, deep learning image analysis models have been more accurate at predicting medical conditions than experts. Deep learning has also been effective for material analysis on photographs. We aim to leverage deep learning to perform material analysis on medical images. Because material datasets for medicine are scarce, we first introduce a texture dataset generation algorithm that automatically samples desired textures from annotated or unannotated medical images. Second, we use a novel Siamese neural network called D-CNN to predict patch similarity and build a distance metric between medical materials. Third, we apply and update a material analysis network from prior research, called MMAC-CNN, to predict materials in texture samples while also learning attributes that further separate the material space. In our experiments, we found that the MMAC-CNN is 89.5% accurate at predicting materials in texture patches, while also transferring knowledge of materials between image modalities.

Keywords: *Research, Neural networks, bioinformatics*

THESIS DUPLICATION RELEASE

I hereby authorize the University of Arkansas Libraries to duplicate this thesis when needed for research and/or scholarship.

Agreed Carson Molder ______________________

Carson Molder

Refused ______________________

Carson Molder

ACKNOWLEDGEMENTS

To begin, I thank my research advisor for the past two years, Professor Justin Zhan. He has been pivotal in introducing me to computer science research and helping me grow the skills I needed to create this thesis. He has always been a strong supporter of my studies, and always offered me opportunities to grow.

I also thank the CSCE Department, College of Engineering, Walton Foundation, and Honors College for their academic and financial support throughout my four years at the University of Arkansas. Their assistance allowed me to attain a world-class education essentially for free. In addition to deepening my computer engineering studies, these organizations have supported my ability to explore many other interesting activities and topics.

The support of my parents and friends has also been a major aspect of my success. By lifting me and validating my efforts, they guided me through my studies, research, and life decisions both before and during college.

Specifically for this thesis, I thank Benjamin Lowe for teaching me many machine learning concepts and being an invaluable partner on this research, Hadi Salman and Alycia Carey for their assistance in developing my approach, and Gabriel Schwartz for providing source code that we referred to during the implementation of this paper.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1 Introduction

Deep learning has recently surged in popularity for use in a broad set of medical applications. The accuracy and automation that deep learning models provide will transform medicine by providing medical experts with powerful tools to effectively diagnose, understand, and treat medical conditions. Many medical problems have already been addressed by deep learning, including annotating pressure wounds [1], segmenting brain MRIs [2], and predicting pneumonia from chest X-ray images [3, 4].

While deep learning has enabled significant medical advances, many of the models proposed eschew interpretability in favor of accuracy. Another challenge with medical deep learning models is acquiring large, high-quality medical datasets, especially for niche problems with few experts who are freely available to annotate new data. This lack of annotators often constrains the set of medical tasks that deep learning models can practically perform.

We believe that evaluating textures and materials in medical images is a task that, while currently challenging, can be made practical with new methods. Material analysis is the procedure of examining local information in images to learn about the physical and textural makeup of the objects present within. It provides an alternative to other popular image analysis methods, like predicting segmentation masks or bounding boxes, and depends only on local context.

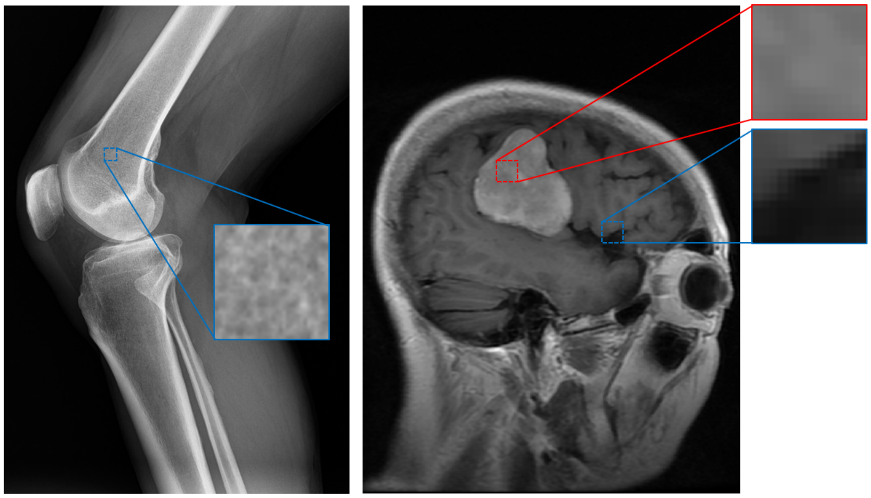

Figure 1.1: Sample texture patches from a knee X-ray and brain MRI.

The major challenge with medical material analysis is a lack of material and textural annotations for medical images. Rather, many medical image datasets are annotated with segmentation masks or bounding boxes. These annotations do not explicitly encode the underlying materials, but rather *regions* of material interest. Figure 1.1 presents an example of such regions of interest, with a knee X-ray and a brain MRI with a tumor.

The insets in Figure 1.1 present examples of interesting textures–namely, bone for the knee X-ray, and tumor (red) and healthy gray matter (blue) for the brain MRI. Notably, these local patches have distinct characteristics. The bone patch appears spongy, while the tumor patch is distinctly brighter than the surrounding gray matter. We believe that it is possible to leverage annotations like segmentation masks that outline these regions of interest, to generate a dataset of local texture patches. Such a texture dataset could help train deep learning models to perform material analysis on medical images.

In this thesis, we propose a novel method to generate these texture patches from higher-level annotations and use these patches to automatically learn a similarity relationship between the generated materials. We then re-implement and update a material analysis neural network from prior research [5] to classify medical materials present in texture patches, while learning a set of attributes that further separate material category predictions on texture patches. Finally, we evaluate this network on full-sized medical radiography images, to determine how well the network can predict materials with local context alone.

The thesis is presented as follows. In Chapter 2, we discuss prior research into computer vision architectures, automatic material analysis, deep learning for biomedical applications, and medical datasets. In Chapter 3, we introduce our method to automatically generate texture patches from annotated and unannotated radiography images. In Chapter 4, we introduce a neural network, D-CNN, that predicts the similarity of texture patches and helps establish a distance metric between different materials. In Chapter 5, we present our adaptation of the method from [5] and introduce MMAC-CNN, a second neural network that predicts medical materials and their related attributes. In Chapter 6, we present our experiments to show the efficacy of our method. Finally, in Chapter 7, we conclude and discuss future work.

2 Related Work

The medical material analysis method proposed in this thesis is inspired by a wide body of research that focuses on computer vision, material analysis, bioinformatics, and medical imaging.

2.1 Computer Vision

Our method uses neural network-based computer vision architectures to make material predictions on local regions of a radiography image. Neural networks have been shown to be effective at performing computer vision tasks like object detection [6], image comparison [7], image generation [8], and video frame interpolation [9]. For classification tasks on image datasets like CIFAR-10/100 [10] and ImageNet [11], multiple advanced neural network architectures have achieved over 90% accuracy [12, 13].

In our method, we use a Siamese neural network model [7] to determine if two image patches are of the same or different material categories. This architecture was chosen because the design has been effective at addressing other similarity-evaluation tasks. These tasks include signature verification [7], human re-identification across multiple cameras [14, 15], object tracking [16, 17], sentence comparison [18], and one-shot image classification [19].

2.2 Material Analysis

Many works have explored the material and textural analysis of images. Our work builds on a weakly supervised method [5] to learn material attributes like softness, roughness, and translucency, from photographs. The method leverages these attributes to make categorical material predictions like “fabric”, “rock”, and “water”. We show that attributes can be leveraged in medical images to increase material prediction accuracy and separate category predictions like “bone” and “tumor”.

Generally, automatic material recognition is implemented by training a model to predict the material category of an annotated image patch sampled from a larger image. Many image patch datasets for materials in natural images have been introduced, including CUReT [20], KTH-TIPS [21, 22], the Flickr Materials Database (FMD) [23], and the Materials in Context (MINC) Database [24]. These datasets have been used as the basis for multiple material prediction models, including a support vector machine (SVM) for KTH-TIPS [21], and convolutional neural networks (CNNs) for FMD and MINC [24, 25, 26].

Image patches have also been adapted for non-material purposes. For instance, the Vision Transformer [13] architecture divides an image into patches for use as a sequential sequence of tokens in a self-attention encoder. To allow these patches to be considered in the context of the entire image, a positional encoding for the patches is learned.

Automated material analysis has also been applied to both broader and domain-specific contexts. For example, prior work [27] has explored learning visual attributes in a weakly supervised manner by mining relevant text information from the internet. Another work [28] applied material analysis to photographs of railroad ties, using bounding boxes that label regions of several domain-specific categories, to detect decaying infrastructure.

2.3 Biomedical Applications

In addition to developments in computer vision and its subfield of material analysis, there have been many applications of deep learning and neural networks to biomedical tasks [29]. These tasks include multimodal heartbeat detection [30], improving ultrasound image quality [31], and making cancer predictions [32].

The broad applicability of Siamese neural networks also extends to medicine. Siamese neural networks have been used for gait recognition [33], spinal metastasis detection [34], brain cytoarchitecture segmentation [35], and disease severity evaluation [36].

One parallel problem to ours is predicting segmentation maps in medical images. U-Net [37] is a CNN-based model that performs this task. Although both U-Net and our method can predict the composition of regions in a medical image, U-Net's architecture directly draws from segmentation maps as the ground truth. Our method instead uses intermediate material representations to achieve this task.

Material analysis methods have also been applied to some specific problems in medicine. For instance, a Bayesian network has been applied to analyze textures in CT scans of liver lesions [38], assessing shape, texture, proximity, and location parameters. Another work [39] leveraged textural features in addition to edge, color, and language descriptors to classify the modality of a medical image (such as an X-ray, MRI, or CT scan).

2.4 Medical Image Data

Our method requires preexisting medical radiography images to generate a texture dataset and perform material analysis. Fortunately, there are many public, annotated medical image datasets for a variety of relevant problems.

For example, CheXpert [3] is a large, public dataset of chest X-rays with detailed annotations. In addition to the dataset, the creators of CheXpert also host a leaderboard of high-scoring models on its dataset, with the best-performing model scoring an AUC of 0.93 [40] as of April 2021. These high scores indicate the data and annotations are rich enough for complex models to make accurate predictions on medical tasks.

High-quality medical image datasets are prevalent in many domains. For example, there are multiple public brain MRI datasets with segmentation masks for brain tumors [41, 42, 43]. In the oncology field alone, image datasets exist for breast cancer mammograms and ultrasounds [44, 45], colon cancer CT scans [46], and lung cancer MRIs and CT scans [47]. Outside of oncology, there are public medical image datasets for hip and knee X-rays [48], Alzheimer's degree progression [49], and tissue microarrays [50]. While we evaluate our method on a relatively narrow application, the breadth of publicly available data makes it practical to apply it to many other problems due to an abundance of imaging data.

3 Texture Patch Generation

To perform medical material analysis using deep learning, we must first define a dataset of medical materials that a model can learn. As demonstrated in Chapter 1, a set of local texture patches with associated material labels is effective at encoding the materials we desire to learn.

The resulting dataset from our dataset generation process is designed to help train a model to predict the associated materials of texture patches in a fully supervised fashion. Therefore, each texture patch is associated with a material label. We call these material labels material categories. Material categories are a set of user-defined labels that capture the set of materials that will be evaluated. For example, an X-ray image may have a “bone” material category, while a brain MRI may have “brain” and “tumor” material categories.

When generating a dataset of texture patches, we allow the user to specify which regions of an image (inside or outside a segmentation mask) should belong to certain material categories. Depending on the image’s relevant annotations and characteristics of a patch, we break down material categories into three types: naïve, expert, and null categories.

The remainder of this chapter is presented as follows. In Section 3.1, we discuss the characteristics of a desirable texture patch dataset. In Section 3.2, we discuss the three types of material categories. In Section 3.3, we discuss steps we take to pre-process and normalize the source medical images. Finally, in Section 3.4, we present our algorithm for generating a texture patch dataset from pre-processed medical images.

3.1 Desired Dataset Properties

Like most tasks in computer vision, having a well-defined problem and dataset for performing material analysis is crucial. While there may be medical material datasets for a variety of problems in the future, these datasets are currently rare or nonexistent. Therefore, we introduce a systematic process for generating material image patches from annotated and unannotated full medical images.

The most significant challenge of generating a dataset of image patches is ensuring that they accurately represent the desired target materials. For a given medical problem, an ideal dataset of image patches would have the following properties:

- **Contains many patches.** Larger datasets yield more information that a model can learn from. If a model can leverage this information, then the additional information from a larger dataset will generally help the model make more accurate material classification decisions.

- **Has correctly sized patches.** Image patches should neither be too large nor too small. If they are too large, the model will become dependent on context to predict underlying textures. If they are too small, relevant local information will be lost.

- **Is derived from expert assessment.** In medicine, most dataset annotations are generated from experts like doctors or nurses. We believe that medical image patches should also be derived from expert annotations. This is especially true in medicine, where computer predictions should be as trustworthy as possible.

- **Categorized appropriately for the problem.** The material categories should reflect the granularity of the problem. For example, predicting brain tumors can be a binary decision ("tumor" or "non-tumor") or focus the specific kind of tumor ("glioblastoma", etc.).

- **Has balanced categories.** Keeping the number of samples in each category balanced makes training a model easier and avoids skewing the predictions. It also ensures that predictions from all categories are similarly trustworthy. If some categories are rare, a large base set of medical images may be needed.

- **Separates patches sampled from the background.** The background has no useful textural information, but excluding background regions from the patch set will lead a model to make arbitrary predictions on these regions, leading to less accurate region boundaries when looking at materials on a full-image level. Assigning patches from these regions to a separate category will let the model consider them separately.

Our patch generation process attempts to create a dataset that has all these properties. On a high level, for each medical image, we preprocess the image, randomly sample patches from it, then assign a label to each patch based on the image's annotations or the average brightness of the patch. We also augment the patch dataset, using transformations like horizontal and vertical reflections, to increase its effective size.

3.2 Types of Material Categories

Our algorithm builds texture patches from three kinds of material categories: naïve categories, expert categories, and null categories, which depend on the patch's brightness and the image's annotations. Some medical material categories, like bone in a knee X-ray, are relatively easy to distinguish in an image, do not need (and often lack) expert annotations, and are the only relevant medical material within an image. In these cases, our algorithm assumes that the brightness of a region corresponds to the material present in that region.

Material categories that fit this case are called naïve categories, because a "naïve" assumption is made that brightness corresponds to a given material. While this analysis is relatively shallow in the context of the same kind of image, we believe a model can use these categories to predict these materials in other kinds of images (for example, brain MRIs) that have more complex material compositions, making those predictions more robust. If a patch comes from an unannotated image and has an average brightness $\bar{B}$ between $\bar{B}_{min}$ and $\bar{B}_{max}$, it is assigned to the naïve category.

Other medical material categories, like brain tumors, require the use of annotations to be separated from other materials of interest, like gray matter and bone. The diagnosis of ailments like brain tumors also requires expert analysis. We call these expertly annotated materials expert categories. Leveraging expert annotations increases the number of medical materials that can be analyzed, while also being more reliable than assigning categories by patch brightness. We define a mask tolerance value T that represents the smallest percentage of a patch that can be inside the mask to be assigned to the in-mask category. If a patch's in-mask percentage is below T, it is assigned to the out-of-mask category.

Finally, we define a null category, where there is little valuable textural information. Including this category allows for better separation of naïve and expert material categories, as these uninformative regions do not correlate with other material categories. The first task of the algorithm, after sampling a patch, is to check whether a patch should be assigned to the null category. The algorithm performs this by assigning all patches with an average brightness $\bar{B}$ less than constraint $\bar{B}_0$ to the null category.

3.3 Image Preprocessing

The first step in our patch generation process is to ensure the images are normalized. Depending on the problem, a variety of image modalities (such as X-rays, MRI scans, CT scans, microscope images, etc.) from different sources may be used. These modalities may yield different image structures (positive or negative contrast, 2D or 3D data, etc.) that need to be accounted for before they can be sampled for material patches.

For 3D images like MRIs, we split them by layers into 2D slices, as our material patches are 2D. This yields more data, as one MRI can have multiple layers. We also invert negatives (images where the brightest regions indicate dark areas). Finally, we normalize the values of the image to the range $[0, 1]$.

3.4 Patch Generation Algorithm

Our patch dataset generation algorithm takes I, a set of preprocessed 2D medical images, with some generation parameters, and yields k sets, $\{C_0, C_1, C_2, ...C_{k-1}\}$, that contain image patches that are labeled as material classes $\{0, 1, 2, ..., k-1\}$ respectively. Algorithm 1 presents the algorithm for generating a texture patch dataset.

For a given image I_i, our algorithm assumes these annotations are segmentation masks M_i around regions of interest. Given a patch comes from an annotated image with a segmentation mask, and its average brightness $\bar{B}$ is between $\bar{B}_{min}$ and $\bar{B}_{max}$, we calculate its in-mask percentage $T(P_i, M_i)$ with Equation 3.2. This value is compared against the mask tolerance constraint T.

The mask tolerance T defines the minimum percentage of a patch that can be inside the segmentation mask to be assigned to the in-mask category set. If a patch is completely within the mask, $T(P_i, M_i) = 1$, and the patch will be added to the in-mask category C_j. If a patch is completely outside the mask, $T(P_i, M_i) = 0$ and the patch will be added to the out-of-mask category C_i. If the patch partially overlaps with the mask, its assignment depends on T. While this algorithm assumes one mask category, it can generalize to multiple masks by assigning a mask tolerance to each mask category and calculating $T(P_i, M_i)$ for each category's segmentation mask.

To ensure our algorithm generates a dataset of exactly N patches, it only increases the counter variable n once a patch is added to a class. If a patch is not added to a class (primarily because the patch's average brightness falls between $\bar{B}_0$ and $\bar{B}_{min}$ or $\bar{B}_{max}$ and 1), then the counter does not increment, and an additional patch is sampled. Therefore, if N is too large, the algorithm may not halt. To mitigate this, we use a cutoff value R that stops sampling patches after R attempts. Generally, R is on the order of 100-1000 times N.

Algorithm 1 The patch label assignment algorithm.

Inputs:

I, set of images
$\bar{B}_{min}$, $\bar{B}_{max}$, minimum and maximum average brightness constraints
$\bar{B}_0$, maximum null-class average brightness cutoff
T, mask tolerance
N, number of patches to generate

Outputs:

C_0, sets of patches with null class label 0
$C_1, C_2, ..., C_{k-1}$, sets of patches with class labels $1, 2$, ..., $k-1$

```
$n = 0$
while $n < N$ do
    Select $I_i$ of class $i$ randomly from $I$
    Select $P_i$ with raw features $p$ randomly from $I_i$
    $\bar{B} = \text{avg}(P_i)$
    if $\bar{B} < \bar{B}_0$ then
        Add $P_i$ to set $C_0$
    else
        if $\bar{B}_{min} \leq \bar{B} \leq \bar{B}_{max}$ then
            if $I_i$ has mask $M_i$ of class $j$ then
                if $T(P_i, M_i) > T$ then
                    Add $P_i$ to set $C_j$
                if $T(P_i, M_i) < (1 - T)$ then
                    Add $P_i$ to set $C_i$
            else
                Add $P_i$ to set $C_i$
    if $P_i$ added to any $C_i$ then
        $n = n + 1$
```

Where:

$$\text{avg}(P_i) = \frac{\sum_{p \in P_i} (p)}{|P_i|} \tag{3.1}$$

$$T(P_i, M_i) = \frac{|\{(x, y) \in P_i | (x, y) \in M_i\}|}{|P_i|} \tag{3.2}$$

4 Discovering Medical Material Similarity

One significant difference between natural image materials and medical materials is that medical materials require expert knowledge to properly discern. Therefore, any reliable metric of similarity between medical materials should be sourced from experts. We care about material similarity because it serves as an analog for the distance between materials. As depicted in Figure 1.1 (Chapter 1), certain pairs of materials, like gray matter and brain tumor, appear more similar to each other than other pairs, like gray matter and bone. By mapping material similarity to a distance metric, we can generate an embedding of the material space that better separates material categories and enhances the training of material classifiers.

Unfortunately, directly relying on experts to generate medical material similarity data makes analyzing medical materials impractical, as doctors are scarce and expensive to retain compared to general image annotators (who could be sourced from crowdsourcing services like Amazon Mechanical Turk [5]).

Despite the scarcity and cost of experts, there are many free, pre-annotated medical images available on the Internet. For example, multiple datasets of brain tumor MRI scans with segmentation masks are freely available [41, 42]. Leveraging such datasets and their pre-made expert annotations to learn the similarity of medical materials would remove the need for additional expert labor and make medical material analysis viable. Although these annotations do not explicitly encode material information, it is reasonable to assume the textural characteristics of the image portions inside these masks differ from the portion outside (as is the case for a brain tumor and healthy gray matter).

In this chapter, we propose a method to leverage these non-material medical annotations to automatically learn medical material similarity, using a novel neural network called D-CNN. In Section 4.1, we present the D-CNN's architecture. In Section 4.2, we present our method to construct a set of patch pair samples from a texture patch dataset. In Section 4.3, we discuss our method to train the D-CNN on a patch pair dataset. Lastly, in Section 4.4, we present our method to generate a material distance metric from the trained D-CNN's positive similarity decisions.

4.1 D-CNN Architecture

D-CNN is a Siamese neural network based on ResNet34 [12]. It receives a reference texture patch (x_r) and a comparison texture patch (x_c) as inputs. First, these patches are passed through a headless ResNet34 network with the same weights. Next, a rectified linear (ReLU) activation function is applied to each resulting hidden state. Then, the hidden states are concatenated and passed through a common linear layer that outputs a prediction on whether the two patches' material categories are the same (s_{similar}) or different ($s_{\text{different}}$). Figure 4.1 depicts the D-CNN architecture. A more detailed version, including all ResNet34 layers, is presented in the Appendix (Figure A.1).

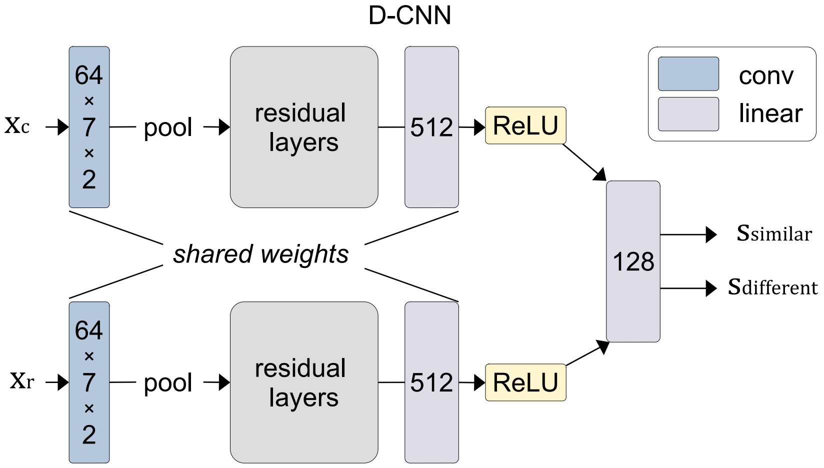

Figure 4.1: The D-CNN model architecture.

The D-CNN's ability to determine the similarity of two texture patches' material categories is based on the common weights and biases in the headless ResNet34 layers that transform the two patches into their hidden states. With common weights, similar appearing texture patches, perhaps from the same material category, should yield similar activations, while different appearing texture patches should yield noticeably different activations.

The final linear layer processes a concatenation of the two patches' hidden states and outputs a similarity prediction. The D-CNN's output can be treated as a one-hot encoding, where $s_{\text{similar}} = 1$ means it predicts that the two texture patches are of the same category, and $s_{\text{different}} = 1$ means it predicts that the two texture patches are of different categories.

The D-CNN makes its predictions independent of the material category labels of the texture patches, meaning it *only* determines similarity, not category. By performing texture patch comparison as a similarity problem instead of a direct classification problem (where each patch's category would be predicted and compared), the D-CNN's problem and solution space is greatly reduced.

We selected ResNet34 as the basis for our convolutional layers over other CNN architectures in the D-CNN for two primary reasons. First, ResNet has well-documented, strong performance on multiple image classification tasks [12]. Second, it is easy to train a deeper ResNet model with extra convolutional layers if a more powerful D-CNN is desired, because the architecture's residual connections address the vanishing gradient problem.

4.2 D-CNN Dataset Construction

To robustly train and evaluate the D-CNN on every possible material category pair, we introduce a patch pair dataset construction algorithm that takes pre-generated texture patches created using the methods in Chapter 3 and groups them into sets of reference (X_r) and comparison (X_c) patches. Algorithm 2 presents the method to construct these sets.

Algorithm 2 The D-CNN dataset construction algorithm.

```
Inputs:
    $\{P_0, P_1, ..., P_{k-1}\}$, sets of texture patches with material category labels $\{0, 1, ..., k-1\}$
    $N$, number of samples for each material category        ▷ By default, $N = \min_y ||P_y||$.
    $k$, number of material categories
Outputs:
    $X_r$, set of reference image patches                     ▷ $||X_r|| = Nk$
    $X_c$, set of sets of comparison image patches            ▷ $\sum_{X_c^{(i)} \in X_c} ||X_c^{(i)}|| = Nk^2$

$X_r = \emptyset$
$X_c = \emptyset$
for $i = 1$ to $k$ do
    $X_r^{(i)} = \text{Sample}(P_i, N)$                        ▷ Sample without replacement
    $X_c^{(i)} = \emptyset$
    for $j = 1$ to $N$ do
        for $l = 1$ to $k$ do
            $x_c^{(l)} = \text{Sample}(P_l, 1)$                ▷ Sample with replacement
        $X_c^{(i,j)} = \{x_c^{(0)}, x_c^{(1)}, ..., x_c^{(k-1)}\}$
        Add $X_c^{(i,j)}$ to $X_c^{(i)}$
    Concatenate $X_r^{(i)}$ to $X_r$
    Concatenate $X_c^{(i)}$ to $X_c$
```

At a high level, Algorithm 2 creates samples of the form $\{x_r, x_c^{(0)}, x_c^{(1)}, ..., x_c^{(k-1)}\}$, where x_r is a reference patch from X_r and each $x_c^{(i)}$ is a comparison patch of category i (out of k material categories), all pulled together from one set in X_c. That is, each reference patch is compared to exactly one patch from each material category. These patches are also associated with their material category labels $\{y_r, y_c^{(0)}, y_c^{(1)}, ..., y_c^{(k-1)}\}$ and similarity labels $\{s^{(0)}, s^{(1)}, ..., s^{(k-1)}\}, s^{(i)} \in \{0, 1\}$, which are generated upon retrieving the sample.

Our dataset also has the following features. First, exactly N reference patches from each material category are generated, ensuring the material categories are balanced in the set. It also follows that there are Nk comparison patches generated for each material category, as each of the Nk entries in the dataset contains one comparison patch from each category. Second, we choose to sample reference patches without replacement and comparison patches with replacement.

We also apply additional methods to improve the dataset's capability to train the D-CNN. During both training and validation, we shuffle the ordering of the samples. During training, we also apply image transformations to the patches (random horizontal and vertical flips) to augment the dataset and effectively increase the sample size.

Since the D-CNN has only two inputs, x_r and x_c, one dataset sample can yield multiple predictions from the D-CNN. For a given sample, we individually pass x_r and each of the $x_c^{(i)}$ patches through the D-CNN up to the ReLU activation unit, and cache their hidden state. Then, we concatenate the hidden state of x_r one-at-a-time to each of the hidden states of the $x_c^{(i)}$ patches, and pass each pair through the final linear layer to receive the predictions for the pairs $(x_r, x_c^{(0)}), (x_r, x_c^{(1)}), ..., (x_r, x_c^{(k-1)})$ against labels $s^{(0)}, s^{(1)}, ..., s^{(k-1)}$ respectively.

4.3 D-CNN Training

To train the D-CNN, we first initialize the convolutional/residual layers with weights pre-trained on ImageNet [11]. The remaining layers are initialized with randomly selected weights. Then, we aim to maximize its accuracy at determining whether two texture patches come from the same or different material categories.

To achieve this, we leverage the texture patches generated using the method from Chapter 3, such that the D-CNN learns from all possible pairs of material categories while avoiding overfitting and class imbalances. Our training methodology also acknowledges that the D-CNN will never perfectly predict the similarity of every pair of material patches—in fact, mispredictions play a role in generating a material distance metric.

To train the D-CNN, we aim to minimize the cross-entropy loss between the D-CNN's similarity predictions and the binary similarity labels. Cross-entropy loss was selected because the D-CNN's output $\hat{s}$ can be interpreted as a probability distribution against the binary similarity labels s. More formally, we present the D-CNN training process as the following minimization objective. Given a D-CNN, represented as function f, with parameters Θ, similarity predictions $\hat{s} = f(x_r, x_c)$, and similarity labels $s \in \{0, 1\}$, the objective for finding optimal parameters Θ^* is presented in Equation 4.1.

$$\Theta^* = \arg\min_{\Theta} \sum_{\{(s,\hat{s})\}} -\left(s \ln(\hat{s}) + (1 - s)\ln(1 - \hat{s})\right) \tag{4.1}$$

By minimizing the cross-entropy loss, the D-CNN will maximize its accuracy and confidence for making similarity predictions on pairs of texture patches.

In addition to our minimization objective, the process for training the D-CNN is also designed to minimize overfitting. For every epoch that the D-CNN is trained on the testing set, its loss is also evaluated on a validation set. The lowest validation loss seen so far of all previous epochs is tracked, and if the current epoch's validation set yields a lower loss than prior epochs, the current D-CNN parameters are saved. By saving the lowest-loss D-CNN instead of the final epoch D-CNN, we mitigate the possibility of loss increasing in later epochs, which would indicate overfitting.

4.4 Encoding Material Distances

The ultimate goal of the D-CNN is to learn a similarity measure between pairs of material categories. This similarity measure can be used to generate an embedding of new attributes that help separate material category predictions and yield underlying information about the materials in the medical material dataset.

To achieve this, we introduce the material distance matrix D, which is generated by analyzing the correctness of the D-CNN on pairs of material categories. To create D, we encode the number of positive similarity decisions for a pair of material categories into a positive decision matrix P, and then build the entries of D from L2-norms of the differences of columns in P.

That is, P is a $k \times k$ matrix, where entry P_{ij} represents the percentage of positive similarity decisions S_{ij} for all reference patches of category i against comparison patches of category j. Equation 4.2 presents the formulation for entry $P_{ij} \in P$.

$$P_{ij} = \frac{1}{N} \sum_{\hat{s} \in S_{ij}} \hat{s} \tag{4.2}$$

Once P is generated, the columns P_i, which represent the similarity decisions for a reference category i, are used to generate matrix D. The entry D_{ij} represents the L2-norm of the difference of columns P_i and P_j. This is presented in Equation 4.3.

$$D_{ij} = ||P_i - P_j||_2 \tag{4.3}$$

Like the D-CNN model, we only keep the material distance matrix D that is generated by the lowest-loss model. That is, we derive the D matrix from similarity decisions on the validation set for each epoch, and only keep the D matrix from the epoch with the lowest validation-set loss.

5 Discovering Medical Material Attributes

While the D-CNN is useful at generating a distance metric for material categories, it does not solve the more relevant challenge of directly predicting the material category of a given texture patch. The D matrix generated by the D-CNN is also limited, as it depends on the patch labels.

To reduce our dependence on labels and improve the separation of material categories in our predictions, we introduce the concept of material attributes, previously presented in [5]. Material attributes are machine-generated parameters that embed relevant information about the material categories. These attributes could implicitly encode relevant properties about materials that category labels do not explicitly portray, like a material's relative brightness. We would like to base our medical material attributes on the material distances in the D matrix, while also considering the ways humans visually perceive materials.

In Section 5.1, we introduce a method to generate an embedding of material attributes that respects the material categories inside the D matrix. Then, in Section 5.2, we introduce MMAC-CNN, a convolutional neural network that improves on work from [5], that can simultaneously classify material categories and attributes in a texture patch. Finally, in Section 5.3, we discuss the minimization objective and training procedure for MMAC-CNN, which allows it to learn how to perform these tasks accurately.

5.1 Generating Material Attributes

While the D matrix generated by the D-CNN is effective at establishing a distance metric between material categories, we are interested in using these distances to establish a relationship between material categories and material attributes. Like matrix D, this metric can serve as an embedding of the material attributes in a low-dimensional space. To achieve this, we reintroduce the method presented in [5] to generate such a metric.

We call this metric the material attribute-category matrix A. A is a $k \times m$ dimensional matrix, where k is the number of material categories as before, and m is the number of material attributes to be encoded. The value m can be chosen at run time. Additionally, the values of A are bounded to the range $A_{ij} \in [0, 1]$, so that its entries can be interpreted as conditional probabilities. For entry A_{ij}, we interpret its value as the conditional probability that a texture patch exhibits attribute j given it is of category i.

The entries of A are found by minimizing an objective that considers the material category distances of D and intuition about the perception of materials. Equation 5.1 presents the optimization objective.

$$A^* = \arg\min_A \left(c(A, D) + \gamma p(A, \alpha, \beta)\right) \tag{5.1}$$

$$c(A, D) = \sum_{(i,j)} (||A_i^T - A_j^T||_2 - D_{ij})^2 \tag{5.2}$$

$$p(A, \alpha, \beta) = \int_0^1 \left(\text{Beta}(x, \alpha, \beta) \ln\left(\frac{\text{Beta}(x, \alpha, \beta)}{q(x, A)}\right)\right) dx \tag{5.3}$$

The optimization objective for optimal matrix A^* is divided into two parts. First, the $c(A, D)$ term captures a distance measure between the A matrix we aim to optimize and D. For all pairs of material categories, represented as indexes $(i, j) \in \{1, 2, ...k\}^2$, we aim to minimize the L2 norm of the difference of rows i and j of A (A_i^T and A_j^T) with respect to entry D_{ij}. This term constrains the optimization process to respect the material category distances generated by D while finding an optimal set of attributes.

The second part of the objective, the $p(A, \alpha, \beta)$ term, tries to constrain the material attributes in A to fit the way humans perceive materials. Like [5], we assume that texture patches from medical materials strongly exhibit one kind of material category, and only weakly exhibit properties of other medical materials. Similar medical materials (like brain tumors and gray matter) will exhibit this relationship less, but we assume it remains present in all material category pairs.

The Beta distribution, with $\alpha = 0.5$ and $\beta = 0.5$, fits this constraint well. Figure 5.1 presents the shape of the Beta distribution. Near the edges of the distribution (0 and 1), the values are large, while near the center, where more similar materials may fall, the values are small.

The $p(A, \alpha, \beta)$ term also considers the embedding of A into a Gaussian kernel density estimate (KDE), notated as $q(x, A)$. Embedding A into a Gaussian KDE allows us to integrate A within the p-term, as it yields a continuous, one-dimensional embedding of the A matrix. The bandwidth parameter, h, is chosen automatically using Scott's Rule [51].

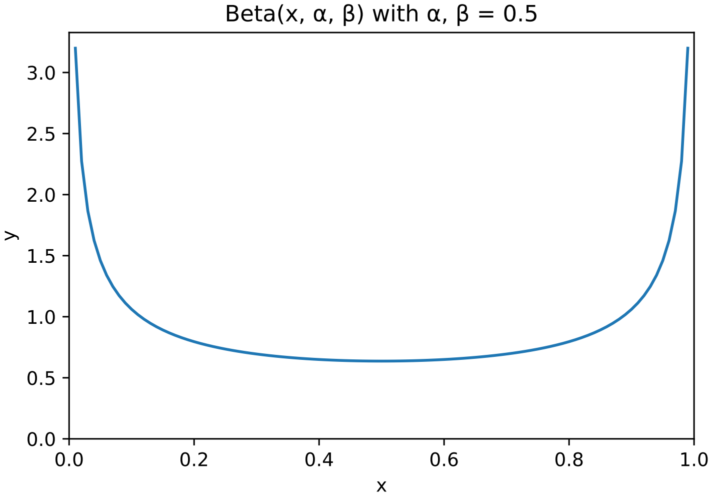

Figure 5.1: A plot of the distribution Beta(x, α, β) with $\alpha, \beta = 0.5$.

The formula for this embedding is presented in Equation 5.4.

$$q(x, A) = \frac{1}{km} \sum_{(i,j)} \left(\sqrt{2\pi h^2} \exp \left(-\frac{(A_{ij} - x)^2}{2h^2} \right) \right) \tag{5.4}$$

Ultimately, $p(A, \alpha, \beta)$ is a Kullback-Leibler (KL) divergence [52] between the Beta distribution and the Gaussian KDE of A. The KL-divergence represents an entropy, or difference, measure between two distributions. In our case, we are trying to minimize the KL-divergence between our ideal Beta distribution and the realized Gaussian KDE embedding of A. We integrate this value over the domain of the Beta distribution ($\mathbb{R} \in [0, 1]$) to yield a total difference measure along the KL-divergence. We also weigh $p(A, \alpha, \beta)$ with hyperparameter γ to influence how much the perceptual distance metric matters during optimization.

In practice, we approximate the integral for $p(A, \alpha, \beta)$ by calculating the values of the Gaussian KDE and Beta distribution at discrete points, and evaluating the points' cumulative entropy. Since the Beta distribution diverges as x approaches 0 and 1, we evaluate the KL-divergence for 25 evenly-spaced points from $x = 0.02$ to $x = 0.98$.

The optimized matrix A^* found using this minimization objective is then used to help the MMAC-CNN to learn material attributes. To estimate the solution to the optimization objective, the L-BFGS-B optimization algorithm [53] is used.

5.2 MMAC-CNN Architecture

The medical material attribute/category convolutional neural network (MMAC-CNN) is a convolutional neural network designed to predict the material attributes and categories present in individual texture patches. MMAC-CNN is based on a similar network from [5], with an updated CNN architecture and a focus on medical materials.

The MMAC-CNN serves as an end-to-end network that can classify the material category of an individual texture patch, while simultaneously predicting the patch's material attributes encoded by the A matrix. It uses convolutional layers from ResNet34 [12], with linear layers designed to predict the material category y of the patch. The MMAC-CNN also contains a set of auxiliary linear layers that yield a probability distribution of the material attributes found in the patch. The MMAC-CNN architecture is presented in Figure 5.2, with a more detailed version presented in the Appendix (Figure A.2).

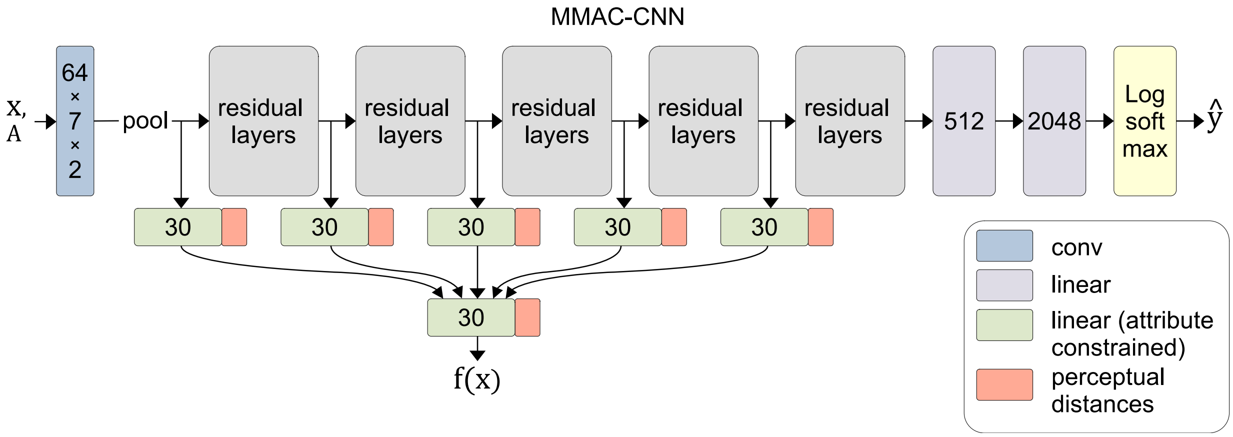

Figure 5.2: The MMAC-CNN model architecture.

Like the D-CNN, we use the ResNet architecture for the MMAC-CNN because of its ability to scale its training to additional layers. This allows MMAC-CNN to easily scale to larger problem spaces (like additional material categories and attributes) and complex medical domains, by adding more residual convolutional blocks to the architecture.

The non-constrained linear layers in the MMAC-CNN are designed to predict the material category of texture patch x. These layers are not constrained as, during training, their weights and biases are not explicitly influenced by the values of the attribute-category matrix A.

The auxiliary linear layers in the MMAC-CNN are designed to yield a probability distribution for material attributes in texture patch x. Each first-layer linear module receives its input from the hidden state of one of the residual layer blocks (or, for the first module, the pooling layer after the initial convolution). A final auxiliary linear layer combines the predictions of the first-layer linear modules, to yield the final probability distribution of material attributes, $f(x)$. The auxiliary network design allows the material attribute predictions to draw information from all levels of the convolutional network, rather than only the final convolutions.

The MAC-CNN model can be interpreted as a function f with parameters Θ that, for a given texture patch x, predicts a probability distribution of the m material attributes ($f(x)$) and k material categories ($\hat{y}$). If the texture patch has d raw features, we write MAC-CNN as $f(x, \Theta) : \mathbb{R}^d \to ([0,1]^m, \mathbb{R}^k)$, where $[0,1]^m$ encodes the material attribute probability distribution and $\mathbb{R}^k$ encodes the material category prediction. For the material attribute probability distribution vector, entry i represents the probability that the patch exhibits material attribute i.

While the sequential layers use a standard rectified linear (ReLU) activation function, with the exception of the final layer, the auxiliary layers employ a capped version of ReLU, $\phi(x)$, that binds the outputs to the range $[0, 1]$. More specifically, we describe the auxiliary layers as a two-layer perceptron with weights and biases $\Theta_{aux} = \{W_1, W_2, b_1, b_2\}$. Given a series of hidden states h, the evaluation of the auxiliary layer is presented in Equation 5.5.

$$\begin{aligned} f(h) &= \phi(W_2\phi(W_1 h + b_1) + b_2) \\ \phi(x) &= \begin{cases} 0 & x \leq 0 \\ x & 0 < x < 1 \\ 1 & x \geq 1 \end{cases} \end{aligned} \tag{5.5}$$

5.3 MMAC-CNN Training

By training the MMAC-CNN, we aim to maximize the network's capability to make accurate material category predictions on individual texture patches, while simultaneously learning a robust set of material attributes, encoded by the A matrix, that enhances the separation of these categories. We also aim to make the training process fast, converging to a low loss in few epochs so that the MMAC-CNN can quickly adapt to new medical problems.

To start, we initialize the MMAC-CNN model with randomly initialized linear layers, and convolutional layers with weights pre-trained on ImageNet [11]. This mirrors the way we initialize the D-CNN and allows a fast convergence to a low loss.

Following the method proposed in [5], we define our loss function in a way that minimizes material category prediction errors, ensures the predictions fit a distribution that mimics human perception of materials, and constrains the material attribute predictor weights with respect to the A matrix. To achieve this, our minimization objective considers three functions, L_c, L_d, and L_a, that consider these three terms respectively.

We present our minimization objective as follows. Given an n-sized batch of texture patches $X = \{x_1, x_2, ..., x_n\}$ with one-hot encoded material category label vectors $Y = \{y_1, y_2, ..., y_n\}$, material category prediction vectors $\hat{Y} = \{\hat{y}_1, \hat{y}_2, ..., \hat{y}_n\}$, and material attribute prediction vectors $f(X) = \{f(x_1), f(x_2), ..., f(x_n)\}$, the minimization objective, which seeks to optimize the MMAC-CNN network parameters Θ, is presented in Equation 5.6.

$$\Theta^* = \arg\min_{\Theta}(L_c + \gamma_d L_d + \gamma_a L_a) \tag{5.6}$$

$$L_c(X, Y) = \sum_{(x_i, y_i)} \sum_{j=1}^{k} -y_{ij} \ln(\hat{y}_{ij}) \tag{5.7}$$

$$L_d(X, \alpha, \beta) = \int_0^1 \left(\text{Beta}(z, \alpha, \beta) \ln \left(\frac{\text{Beta}(z, \alpha, \beta)}{q(z, f(X))} \right) \right) dz \tag{5.8}$$

$$L_a(X, Y, A) = \sum_{c=1}^{k} \left\| A_c^T - \frac{1}{|X_c|} \sum_{i=1}^{|X_c|} f(x_i) \right\|_2^2 \tag{5.9}$$

At a high level, our minimization objective is a weighted sum of three functions—the material category loss function L_c, the material attribute KL-divergence function L_d, and the material attribute constraint function L_a. The latter two functions are weighted by hyperparameters γ_d and γ_a respectively. These parameters can be chosen at training time.

The first component of the minimization objective, L_c, is presented in Equation 5.7. L_c is the negative log-likelihood loss between the material category prediction $\hat{y}$ and the material category label y. For each given sample (x_i, y_i), the loss only considers the network's prediction for the correct label, as y_i is one-hot encoded, and has a value of 1 in one entry and 0 elsewhere. Therefore, any entry in $\hat{y}_i$ outside the correct label will be multiplied by zero and ignored. These losses are summed over the entire sample set to yield the unweighted L_c term.

The second component of the minimization objective, L_d, is presented in Equation 5.8. Similar to the minimization objective for the A matrix, we aim to constrain the MMAC-CNN's material attribute predictions to fit the way humans perceive materials. The L_d term achieves this by calculating the KL-divergence of the Gaussian KDE embedding of the MMAC-CNN's attribute predictions, $q(z, f(X))$, and a Beta distribution, $\text{Beta}(z, \alpha, \beta)$. Like the A matrix objective, we hold $\alpha, \beta = 0.5$. In practice, we approximate the integral by sampling points of the function, similar to the KL-divergence term in the A matrix minimization objective.

The third component of the minimization objective, L_a, is presented in Equation 5.9. The goal of this component is to constrain the auxiliary classifiers in the MMAC-CNN to respect the material attribute-category matrix A. For each material category c, a mean squared error between row c of A and the average $f(x_i)$ prediction for the subset of patches of category c (denoted X_c) is calculated. As this result is a vector, we normalize it to the squared L2 norm. Finally, over all material categories, these normalized values are summed.

The training procedure for the MMAC-CNN employs two methods to reduce overfitting. First, like the D-CNN, the MMAC-CNN training procedure saves the model with the lowest loss on the validation set of patches. This avoids the potential of saving a model in the final epoch with greater loss due to the model losing its ability to generalize training set features to the validation set.

Second, unlike the D-CNN, the MMAC-CNN employs a learning rate scheduler that reduces the learning rate by a factor of 10 after an epoch that yields an increase in validation set loss. This regularizes the learning process and makes it more predictable, further mitigating the potential of overfitting in later epochs.

6 Experiment

To demonstrate the effectiveness of our method for analyzing medical materials, we implemented our design and evaluated it on a dataset of knee X-rays and brain MRIs with tumors. In this chapter, we discuss our implementation, experiments, and results.

The patch generation algorithm, D-CNN neural network, material attribute/category matrix optimization algorithm, and MAC-CNN neural network were all implemented using Python. Specifically, we used the PyTorch neural network library and SciPy scientific computing library to implement the networks and equations used in our method. This implementation can be found at the following GitHub repository: `https://github.com/cmolder/medical-materials`. The majority of our experiments, unless otherwise noted, were run on a desktop computer with an Intel Core i9 processor and two Nvidia Quadro RTX 8000 graphic cards.

In Section 6.1, we describe the specific medical image datasets used to evaluate our models. In Section 6.2, we evaluate the training methodology proposed for the D-CNN and MMAC-CNN and describe its effects on the resulting models' loss and accuracy on our dataset. In Section 6.3, we evaluate the D-CNN's ability to predict material similarity, and the generation of the distance matrix D. In Section 6.4, we discuss our findings from generating the A matrix and evaluating the MMAC-CNN's ability to predict material categories and attributes. Finally, in Section 6.5, we apply the trained MMAC-CNN network in a sliding-window manner to full medical images to see how medical material categories and attributes can be applied to larger contexts.

6.1 Dataset Generation

We source our texture patches from three medical datasets, generating four material categories: "brain", "tumor", "bone", and "background" (the null category). To generate the "brain" and "tumor" category texture patches, we source brain MRIs from two datasets: Brain-Tumor-Progression [42, 54] (3804 images) and brain tumor dataset [41] (918 images). These datasets, stored as DICOM images and MATLAB matrices respectively, contain three-dimensional MRI scans of patients with brain tumors. These three-dimensional scans contain volumetric masks around the regions of each image that contain a brain tumor. We map this data to our dataset by treating each layer and its mask as an individual image to sample patches from.

To generate the “bone” category texture patches, we source knee X-rays from the Cohort Hip and Cohort Knee (CHECK) dataset [48] (300 images). This dataset contains metadata that tracks the onset of osteoarthritis in the knees and hips of patients in the Netherlands. However, we disregard this metadata and only consider the images themselves. We apply this dataset to determine if the method can distinguish bone from the materials found in brain MRIs. Because these images lack masks around the bone region, we instead use brightness constraints and assume that bright regions of each X-ray should be considered bone.

The patch generation procedure also generates background patches from the darkest regions (average normalized brightness < 0.05) of these images. These images are separated from the other categories and predicted separately to make predictions for out-of-mask categories (“brain” and “bone”) more precise.

Once all categories are generated, we separate the final patch set into training, validation, and testing sets on a randomly assigned 60-20-20 split. To generate these categories, we set the sample count, brightness constraints, and mask tolerance constraints as presented in Table 6.1.

Dataset	N	$\bar{B}_{min}$	$\bar{B}_{max}$	$\bar{B}_0$	T
Brain-Tumor-Progression	10000	0.25	1.0	0.05	0.1
brain tumor dataset	30000	0.25	1.0	0.05	0.1
CHECK	13525	0.25	0.85	0.05	n/a

Table 6.1: The constraints used to generate the texture patch dataset.

6.2 Training Evaluation

To ensure that the D-CNN and MMAC-CNN classifiers are effective at performing medical material analysis tasks, we implement and evaluate robust training schemes for each network. Specifically, we implement the training schemes proposed for the D-CNN in Section 4.3 and the MMAC-CNN in Section 5.3, using the parameters presented in Table 6.2.

Model	Notation	Definition	Value
D-CNN	E	Number of epochs	15
	B	Batch size	50
	η	Learning rate	10^{-3}
MMAC-CNN	E	Number of epochs	15
	B	Batch size	50
	η_0	Initial learning rate	10^{-3}
	γ_d	KL-divergence weight	10^{-2}
	γ_a	A constraint weight	1

Table 6.2: The D-CNN and MMAC-CNN training parameters.

For both the D-CNN and MMAC-CNN, we wanted to evaluate whether the training process yields weights that overfit to the training dataset. Overfitting occurs when a model adapts its weights to the training set so strongly that it can no longer generalize its predictions to other samples like a validation set. While the convolutional layers of both models are initialized to a known state, as they are pre-trained on ImageNet [11], the linear layers have random initializations that may impact the training process. Therefore, we evaluate a group of models, as a single instance may not represent the general results of training.

We also evaluated each model's loss and accuracy over distributions of 30 models on the training and validation sets. We define accuracy for the D-CNN as the percentage of correct similarity decisions, and for the MMAC-CNN as the percentage of correct material category predictions. These distributions are plotted with respect to each epoch in the training process. Figure 6.1 presents these distributions. The lines represent the median loss and accuracy, while the shaded regions represent the region between the 25th and 75th percentiles of loss and accuracy. We also present the results from one typical sample in the Appendix (Figure A.3).

Figure 6.1 demonstrates that both the D-CNN and MMAC-CNN tend to have a lower loss and higher accuracy as the training process progresses. However, the training process for the MMAC-CNN is significantly more effective at reducing loss and increasing accuracy. We believe that this can be partially attributed to the learning rate scheduler discussed in Section 5.3.

While the D-CNN loss distribution shows overfitting in later epochs, we mitigate this by saving the lowest-loss model. By saving the lowest-loss model instead of the final epoch model, we guarantee that the best-performing model during training is kept. This tactic is also employed in the MMAC-CNN's training process, but it typically converges to its lowest

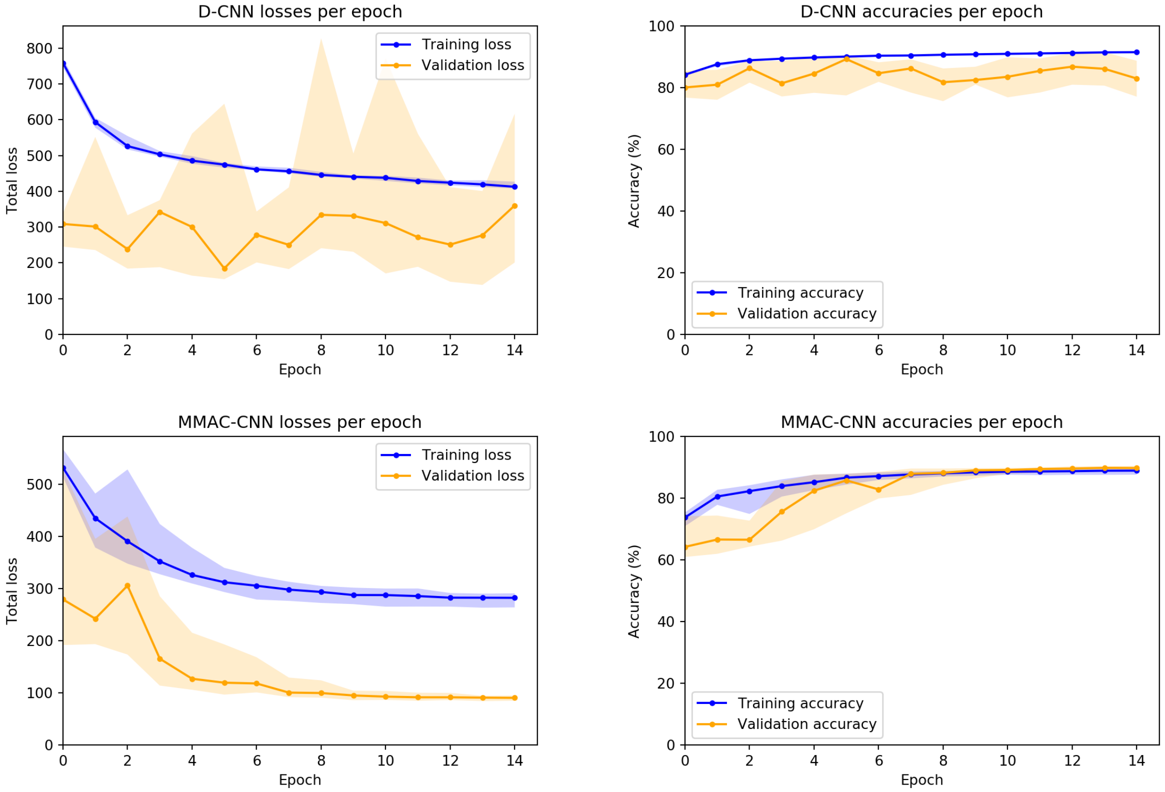

Figure 6.1: D-CNN and MMAC-CNN loss and accuracy distributions per epoch, for 30 trained networks.

loss near or during the final epoch. It may be possible to regulate the D-CNN training process by using a learning rate scheduler or starting with a lower learning rate.

We also consider the amount of time required to train the D-CNN and MMAC-CNN. We would like these models to have a short training time so they can be quickly deployed in new medical fields. To evaluate this, we time the training process for ten instances of the D-CNN and MMAC-CNN. Our results show that the training time for both models, starting with the pre-trained ImageNet layers, is relatively short.

We timed the models' training processes on a separate machine with an Intel Core i7 processor and a single Nvidia GeForce RTX 2080 Ti graphics card. Training 10 D-CNN models with our 15-epoch training process required **23.7 minutes** ± 6.1 seconds per instance. Meanwhile, training 10 MMAC-CNN models with our 15-epoch training process required **14.3 minutes** ± 8.6 seconds per instance.

6.3 Material Categories and D-CNN

To evaluate D-CNN, we trained it on the training and validation sets generated from our three image sources, and then evaluated it on a separate testing set. Our testing set contained 42,768 patches with evenly split categories. On this testing set, the D-CNN was **90.79%** accurate at determining whether a reference and comparison patch were of the same or different material categories.

To more robustly evaluate the D-CNN's performance, we also analyzed the accuracy of the network on each pair of material categories. Table 6.3 presents the accuracy for each pair of categories.

Ref \ Comp	background	tumor	bone	brain
background	100.0	100.0	100.0	100.0
tumor	100.0	53.0	96.2	70.9
bone	100.0	95.9	99.0	98.9
brain	100.0	71.7	98.7	72.6

Table 6.3: The D-CNN accuracy percentages for each pair of material categories.

The accuracy table shows that the D-CNN is more effective at determining similarity for some material categories. Namely, the D-CNN is perfect at determining similarity when one of the patches is from the "background" category. This is likely because the "background" category is visually distinct and homogeneous. The background patches are entirely built from the dark backgrounds of the patch set's source images.

For some pairs of non-background categories, the D-CNN is quite accurate at discerning similarity. Specifically, the D-CNN is effective at separating "brain" and "tumor" patches from "bone" patches. It is also effective at determining if two patches share the "bone" category. This implies that bone is a distinct material from brain tissue and the image background, and, when generating a distance metric, should have a greater distance from the other materials.

The D-CNN is less accurate at discerning between "brain" and "tumor" patches. It especially struggles when predicting whether two "tumor" patches are similar. This implies that tumors are heterogeneous and take on a broad array of texture appearances. This also suggests that discerning a brain tumor benefits from the added context of the whole MRI. Doctors can discern brain tumors from how they contrast with the brain's healthy gray matter. Meanwhile, the D-CNN has better accuracy when predicting similarity among "brain" patches, suggesting that healthy gray matter has a more homogeneous texture.

We also note that the accuracy matrix is not symmetric. This means that either the final linear layer takes some consideration into each patch's position as the reference or comparison patch, or there is some slight variation among the reference and comparison patch sets. While this is not ideal behavior, the difference is small enough (within 0.5%) that it has a negligible impact on the classifier's predictions.

The D-CNN's positive similarity decisions are used to generate the material distance matrix D. In Table 6.4, we present one example of this distance metric generated from the lowest-loss D-CNN on the testing set.

	background	tumor	bone	brain
background	0	1.287	1.376	1.401
tumor	1.287	0	1.004	0.236
bone	1.376	1.004	0	1.231
brain	1.401	0.236	1.231	0

Table 6.4: A sample D matrix generated by the D-CNN's similarity decisions.

We see that the accuracies above, as well as the intuition deduced from these accuracies, translate to the distance metric. For example, the "background" category is relatively far from the other categories, while the "brain" and "tumor" categories have a short distance. The D matrix is also symmetric and intuitively encodes that a category's distance from itself should be zero. In the following experiment, we use the distances between categories in the D matrix to encode a set of material attributes.

6.4 Material Attributes and MMAC-CNN

To begin, we generate A from the D matrix generated by the D-CNN, using the L-BFGS-B algorithm [53] and the method in Section 5.1. We configured the A matrix to have $m = 4$ material attributes and set the KL-divergence weight parameter to $\gamma = 0.1$. We define the "optimal" A matrix as having the lowest loss term $d(D, A)$. While the loss measure can vary significantly over many runs of the algorithm, the most optimal A matrix we generated had a loss $d(D, A) = 1.18$. This is the matrix we used to train the MMAC-CNN for the following experiments.

First, we trained an instance of the MMAC-CNN using the procedure in Section 5.3. On our testing set, this model attained an accuracy of **89.49%** for classifying the material category of a given texture patch. For comparison, the analogous MAC-CNN described in [5] attained an accuracy of 84% for one material category on a photograph patch set with 13 categories. We believe our higher accuracy can be attributed to the ResNet architecture's ability to train more effectively and because we are examining a smaller problem space. Our network only evaluates 4 categories, while [5] evaluated 13.

Second, we determined how our custom loss function and auxiliary network impact the MMAC-CNN's capability to learn to predict material categories. To evaluate this, we withheld the material attributes and calculated loss as simply the mean squared error between the material category predictions and labels. When evaluating this version of the MMAC-CNN on the testing set, the model attained an accuracy of **90.51%**. This suggests that our custom loss function and the evaluation of material attributes have a negligible impact on the MMAC-CNN's ability to predict material categories.

Third, we evaluated the relationship between the material categories and attributes predicted by the MMAC-CNN. We computed the cross-correlation for every category and attribute prediction by the MMAC-CNN on the testing set. These cross-correlations were then embedded in a correlation matrix, which we present in Table 6.5.

Cat Attr	background	tumor	bone	brain
0	+0.45	-0.93	-1.0	-0.85
1	+0.48	-0.01	-0.02	+0.03
2	+0.46	+0.57	+0.52	+0.57
3	+0.51	-0.25	+0.57	-0.22

Table 6.5: The correlation matrix for MMAC-CNN category and attribute predictions.

The correlation matrix demonstrates some relationships among the material attributes and categories. First, the "background" category tends to have a slight positive correlation with every attribute. This seems abnormal, but the background category's feature space is homogeneous, so it makes sense that it does not have a strong bias towards any attribute. Meanwhile, "tumor", "bone", and "brain" patches are correlated with attribute 2.

From the view of the attributes, attribute 2 tends to pick out non-background regions of the image. Meanwhile, attribute 3 tends to correlate strongly with both the background and bone regions of an image. Noticeably, attribute 0 is strongly negatively correlated to non-background regions of the image, and attribute 1 does not correlate strongly to any particular category.

While the correlation matrix establishes a relationship between material attributes and categories, we also determine how the attributes separate the material category space among texture patches. To achieve this, we compare the t-SNE embeddings [55] of the MMAC-CNN's predicted material attribute distributions against the raw features of the texture patches. The t-SNE embedding is a useful metric for evaluating a high-dimensional space, as it separates the distance between samples in a way that preserves their differences—more dissimilar predictions and features will appear farther apart on the graph.

We present the t-SNE embedding of the texture patch raw features and MMAC-CNN material attribute prediction distributions in Figure 6.2. While the raw features can effectively separate background patches, the embedding shows that they are only marginally effective at separating other categories. Meanwhile, the MMAC-CNN material attribute predictions effectively separate all four material categories and preserve their intuitive distances. For example, the prediction embedding yields a relatively short distance for "brain" and "tumor" texture patches. Meanwhile, "bone", a more unique category, is clustered separately in the graph.

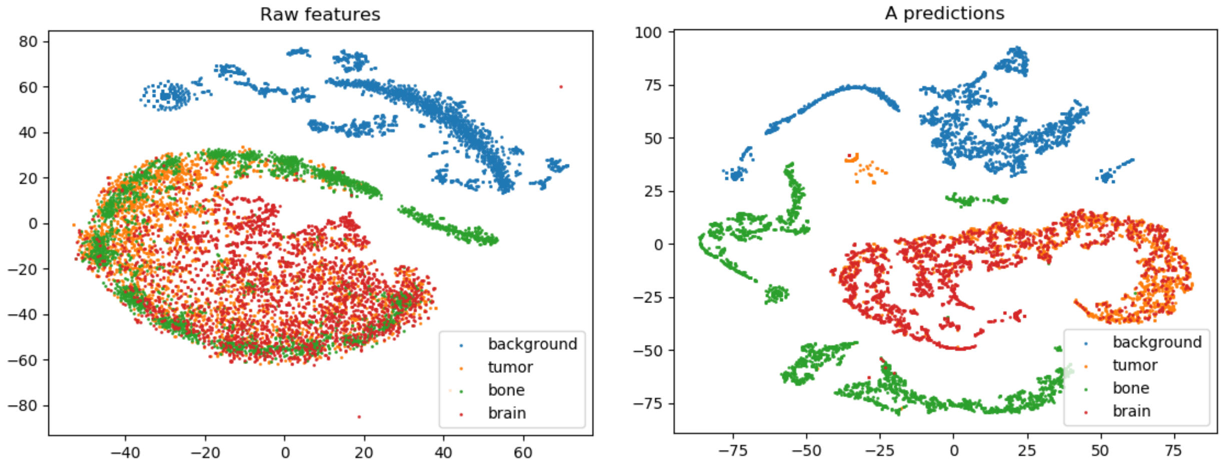

Figure 6.2: The t-SNE embedding of the texture patch raw features and the MMAC-CNN attribute predictions.

6.5 Materials in Full Medical Images

In our final set of experiments, we apply the MMAC-CNN's material category and attribute predictions to full medical radiography images. While we have already shown that the MMAC-CNN is effective at predicting these categories and attributes on local texture patches, we would like to see how these local predictions perform in a greater scope.

To achieve this, we run the MMAC-CNN on a sliding window over each full medical image. We iteratively sample 32 × 32 pixel texture patches over each full medical image with a stride of 4 pixels, recording the MMAC-CNN's category and attribute predictions on each patch. Since the locations of these patches are known, we map the predictions to their location and interpret the predictions as an overlay on the original image.

We performed these predictions on a set of brain MRIs reserved separately from the patch generation procedure and mapped both the category and attribute predictions onto this set of images. In Figure 6.3, we present the MMAC-CNN's category predictions over a set of 4 brain MRIs. The first column contains the base image and mask, and the second column contains the material category predictions. Additional samples are presented in the Appendix (Figure A.4).

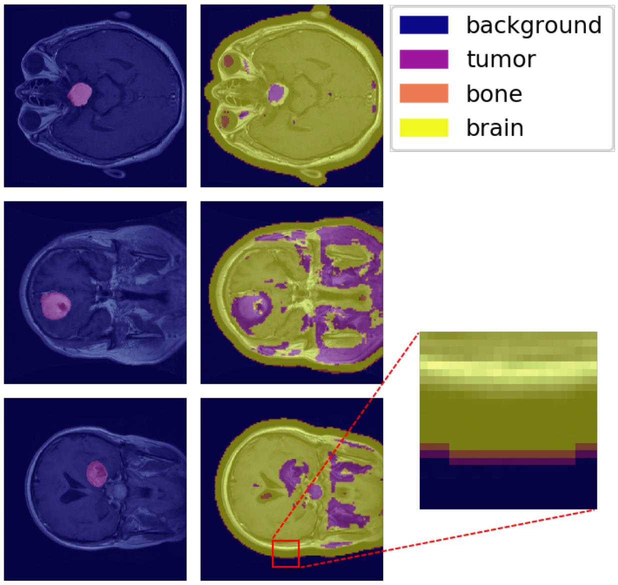

Figure 6.3: Samples from evaluating MMAC-CNN category predictions on brain MRIs.

The full-image material category predictions yield some interesting findings. First, we find that the MMAC-CNN can identify in-mask "tumor" patches effectively—in each sample, the region with the tumor was correctly predicted. However, the MMAC-CNN appears to be sensitive to the "tumor" category, as it predicts tumors elsewhere in the image. This may be due to the relative similarity between healthy gray matter and brain tumors. Additionally, this could be due to a lack of broader context, as the MMAC-CNN only considers individual patches.

We also find that the MMAC-CNN performs knowledge transfer from the knee X-rays. As shown in the inset, the perimeter of the skull tends to pick up predictions of "bone" (colored orange) from the network, even though this category was never sampled from brain MRIs during the patch generation process. This highlights a major advantage of our method—because the training is context-agnostic, materials found in one image modality can be predicted in other modalities without needing to modify the network.

In addition to the material categories, we assess the MMAC-CNN's attribute prediction distributions for the $m = 4$ material attributes it learned. In Figure 6.4, we present the MMAC-CNN's attribute predictions for each of the 4 material attributes on one brain MRI. On the left, we have the base image and mask. The center column contains the attributes overlaid onto the original image, while the right column contains a heatmap for each of the four attributes. Additional samples are presented in the Appendix (Figure A.5).

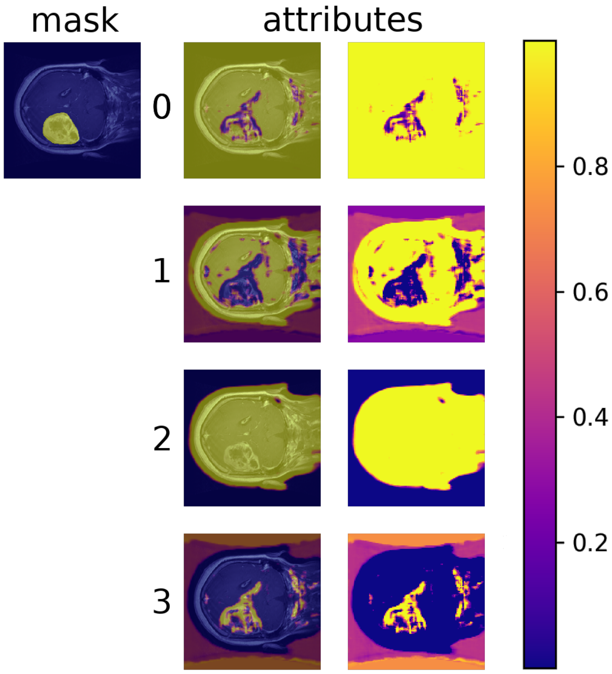

Figure 6.4: Sample from evaluating MMAC-CNN attribute predictions on brain MRIs.

Each attribute tends to pick up on different information in a brain MRI. For instance, as implied by the correlation matrix in Table 6.5, attribute 3 tends to respond highly to tumors. Meanwhile, attribute 2 responds to non-background regions of the MRI, attribute 1 responds to healthy gray matter, and attribute 0 responds to non-tumor regions. Applying the attribute predictions to full brain MRI images solidifies the relationships presented in the MMAC-CNN's category-attribute prediction correlations.

7 Conclusion

The analysis of medical materials is valuable because it enables computers to automatically probe low-level information about medical images. We found that our method, which uses the D-CNN material similarity network and MMAC-CNN material classifier, was effective at predicting medical materials in local regions of medical images. Further, we found that the use of material attributes, explored in [5], translates well to the medical domain, and allows for a better separation of medical categories.

Our method was effective at learning a range of material categories, from healthy gray matter, brain tumors, and bone, and can robustly pick up on a subset of these categories without needing *a priori* knowledge of the modality or image subject. We also found some interesting evidence of knowledge transfer between two modalities, knee X-rays and brain MRIs, that implies our method can easily be extended to other medical problems with few modifications.

Overall, the D-CNN and MMAC-CNN effectively perform material analysis using only pre-existing, non-textural image annotations. Our system avoids the need for the manual generation of texture annotations by domain experts, and also follows humans' high-level intuition about the similarity of materials we find in medical images.

We believe there are multiple ways to improve our method to further enhance its material analysis capabilities. First, we believe the method can be tuned to perform more granular material analysis by leveraging more detailed annotations. For example, training the D-CNN and MMAC-CNN on patches generated from brain MRI data with many annotations could yield greater accuracy and less noise than our broad approach of classifying tumors in general. This could be especially useful when applied to full-sized medical images.

Second, we believe that incorporating broader context into the material predictions could yield more accurate results for more general classification tasks. As discussed in our experiment, we believe many categories of medical materials, like brain tumors, are perceived within surrounding context. For full-sized medical images, we could implement a non-fixed sliding window approach and follow the method proposed in U-net [37], at the cost of requiring complete segmentation maps for each image during training.

Third, we could adapt our method to respect the structure of some radiography images. For instance, MRIs are stored as three-dimensional volumetric images, but we only analyze 2D slices of this data. We believe that our method could generalize to three-dimensional data and learn these kinds of categories in a volumetric manner. Additionally, we could also explore the progression of a subject's medical images over time, which would be useful for fields like oncology where the success of cancer treatments is often visually tracked through periodic imaging.

Bibliography

[1] S. Zahia, D. Sierra-Sosa, B. Garcia-Zapirain, and A. Elmaghraby, “Tissue classification and segmentation of pressure injuries using convolutional neural networks,” *Computer Methods and Programs in Biomedicine*, vol. 159, pp. 51 – 58, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0169260717314864

[2] J. Lai, H. Zhu, and X. Ling, “Segmentation of brain mr images by using fully convolutional network and gaussian mixture model with spatial constraints,” *athematical Problems in Engineering*, vol. 2019, 2019. [Online]. Available: https://www.hindawi.com/journals/mpe/2019/4625371/

[3] J. Irvin, P. Rajpurkar, M. Ko, Y. Yu, S. Ciurea-Ilcus, C. Chute, H. Marklund, B. Haghgoo, R. Ball, K. Shpanskaya *et al.*, “Chexpert: A large chest radiograph dataset with uncertainty labels and expert comparison,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 590–597.

[4] B. Wang, Z. Wu, Z. U. Khan, C. Liu, and M. Zhu, “Deep convolutional neural network with segmentation techniques for chest x-ray analysis,” in *2019 14th IEEE Conference on Industrial Electronics and Applications (ICIEA)*, June 2019, pp. 1212–1216.

[5] G. Schwartz and K. Nishino, “Recognizing material properties from images,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 8, pp. 1981–1995, 2020.

[6] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[7] J. Bromley, J. W. Bentz, L. Bottou, I. Guyon, Y. LeCun, C. Moore, E. Säckinger, and R. Shah, “Signature verification using a “siamese” time delay neural network,” *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 7, no. 04, pp. 669–688, 1993.

[8] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial networks,” *arXiv preprint arXiv:1406.2661*, 2014.

[9] W. Bao, W.-S. Lai, C. Ma, X. Zhang, Z. Gao, and M.-H. Yang, “Depth-aware video frame interpolation,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3703–3712.

[10] A. Krizhevsky, G. Hinton *et al.*, “Learning multiple layers of features from tiny images,” 2009.

[11] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “Imagenet: A large-scale hierarchical image database,” pp. 248–255, 2009.

[12] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[13] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, “An image is worth 16x16 words: Transformers for image recognition at scale,” *arXiv preprint arXiv:2010.11929*, 2020.

[14] R. R. Varior, M. Haloi, and G. Wang, “Gated siamese convolutional neural network architecture for human re-identification,” in *European conference on computer vision.* Springer, 2016, pp. 791–808.

[15] D. Chung, K. Tahboub, and E. J. Delp, “A two stream siamese convolutional neural network for person re-identification,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 1983–1991.

[16] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr, “Fully-convolutional siamese networks for object tracking,” in *European conference on computer vision.* Springer, 2016, pp. 850–865.

[17] Q. Guo, W. Feng, C. Zhou, R. Huang, L. Wan, and S. Wang, “Learning dynamic siamese network for visual object tracking,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 1763–1771.

[18] J. Mueller and A. Thyagarajan, “Siamese recurrent architectures for learning sentence similarity,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 30, no. 1, 2016.

[19] G. Koch, R. Zemel, and R. Salakhutdinov, “Siamese neural networks for one-shot image recognition,” in *ICML deep learning workshop*, vol. 2. Lille, 2015.

[20] K. J. Dana, B. Van Ginneken, S. K. Nayar, and J. J. Koenderink, “Reflectance and texture of real-world surfaces,” *ACM Transactions On Graphics (TOG)*, vol. 18, no. 1, pp. 1–34, 1999.

[21] E. Hayman, B. Caputo, M. Fritz, and J.-O. Eklundh, “On the significance of real-world conditions for material classification,” in *European conference on computer vision.* Springer, 2004, pp. 253–266.

[22] B. Caputo, E. Hayman, and P. Mallikarjuna, “Class-specific material categorisation,” in *Tenth IEEE International Conference on Computer Vision (ICCV’05) Volume 1*, vol. 2, 2005, pp. 1597–1604 Vol. 2.

[23] L. Sharan, R. Rosenholtz, and E. Adelson, “Material perception: What can you see in a brief glance?” *Journal of Vision*, vol. 9, no. 8, pp. 784–784, 2009.

[24] S. Bell, P. Upchurch, N. Snavely, and K. Bala, “Material recognition in the wild with the materials in context database,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[25] L. SIfre and S. Mallat, “Rigid-motion scattering for texture classification,” 2014.

[26] M. Cimpoi, S. Maji, and A. Vedaldi, “Deep filter banks for texture recognition and segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3828–3836.

[27] T. L. Berg, A. C. Berg, and J. Shih, “Automatic attribute discovery and characterization from noisy web data,” in *European Conference on Computer Vision.* Springer, 2010, pp. 663–676.

[28] X. Gibert, V. M. Patel, and R. Chellappa, “Material classification and semantic segmentation of railway track images with deep convolutional neural networks,” in *2015 IEEE International Conference on Image Processing (ICIP)*, 2015, pp. 621–625.

[29] A. Esteva, K. Chou, S. Yeung, N. Naik, A. Madani, A. Mottaghi, Y. Liu, E. Topol, J. Dean, and R. Socher, “Deep learning-enabled medical computer vision,” *npj Digital Medicine*, vol. 4, no. 1, pp. 1–9, 2021.

[30] M. R. Schwob, A. Dempsey, F. Zhan, J. Zhan, and A. Mehmood, “Robust multimodal heartbeat detection using hybrid neural networks,” *IEEE Access*, vol. 8, pp. 82 201–82 214, 2020.

[31] M. Dai, S. Li, Y. Wang, Q. Zhang, and J. Yu, “Post-processing radio-frequency signal based on deep learning method for ultrasonic microbubble imaging,” *Biomedical engineering online*, vol. 18, no. 1, pp. 1–19, 2019.

[32] G. Campanella, M. G. Hanna, L. Geneslaw, A. Miraflor, V. W. K. Silva, K. J. Busam, E. Brogi, V. E. Reuter, D. S. Klimstra, and T. J. Fuchs, “Clinical-grade computational pathology using weakly supervised deep learning on whole slide images,” *Nature medicine*, vol. 25, no. 8, pp. 1301–1309, 2019.

[33] C. Zhang, W. Liu, H. Ma, and H. Fu, “Siamese neural network based gait recognition for human identification,” in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE, 2016, pp. 2832–2836.

[34] J. Wang, Z. Fang, N. Lang, H. Yuan, M.-Y. Su, and P. Baldi, “A multi-resolution approach for spinal metastasis detection using deep siamese neural networks,” *Computers in biology and medicine*, vol. 84, pp. 137–146, 2017.

[35] H. Spitzer, K. Kiwitz, K. Amunts, S. Harmeling, and T. Dickscheid, “Improving cytoarchitectonic segmentation of human brain areas with self-supervised siamese networks,” in *International Conference on Medical Image Computing and Computer-Assisted Intervention.* Springer, 2018, pp. 663–671.

[36] M. D. Li, K. Chang, B. Bearce, C. Y. Chang, A. J. Huang, J. P. Campbell, J. M. Brown, P. Singh, K. V. Hoebel, D. Erdoğmuş *et al.*, "Siamese neural networks for continuous disease severity evaluation and change detection in medical imaging," *NPJ digital medicine*, vol. 3, no. 1, pp. 1–9, 2020.

[37] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention.* Springer, 2015, pp. 234–241.

[38] N. B. Marvasti, E. Yörük, and B. Acar, "Computer-aided medical image annotation: Preliminary results with liver lesions in ct," *IEEE Journal of Biomedical and Health Informatics*, vol. 22, no. 5, pp. 1561–1570, 2018.

[39] I. Dimitrovski, D. Kocev, I. Kitanovski, S. Loskovska, and S. Džeroski, "Improved medical image modality classification using a combination of visual and textual features," *Computerized Medical Imaging and Graphics*, vol. 39, pp. 14–26, 2015.

[40] Z. Yuan, Y. Yan, M. Sonka, and T. Yang, "Robust deep auc maximization: A new surrogate loss and empirical studies on medical image classification," 2020.

[41] J. Cheng, "brain tumor dataset," 2017. [Online]. Available: https://figshare.com/articles/brain_tumor_dataset/1512427/5

[42] K. Schmainda and M. Prah, "Data from brain-tumor-progression," 2018.

[43] S. Bakas, H. Akbari, A. Sotiras, M. Rozycki, and J. Kirbi, "Segmentation labels and radiomic features for the pre-operative scans of the tcga-gbm collection," 2017.

[44] M. Lopez, N. Posada, D. C. Moura, R. R. Pollán, J. M. F. Valiente, C. S. Ortega, M. Solar, G. Diaz-Herrero, I. Ramos, J. Loureiro *et al.*, "Bcdr: a breast cancer digital repository," in *15th International conference on experimental mechanics*, vol. 1215, 2012.

[45] F. M. Calisto, N. Nunes, and J. C. Nascimento, "Breastscreening: On the use of multimodality in medical imaging diagnosis," in *Proceedings of the International Conference on Advanced Visual Interfaces*, ser. AVI '20. New York, NY, USA: Association for Computing Machinery, 2020.

[46] K. Smith, K. Clark, W. Bennett, T. Nolan, J. Kirby, M. Wolfsberger, J. Moulton, B. Vendt, and J. Freymann, "Data from ct_colonography," 2015.

[47] National Cancer Institute Clinical Proteomic Tumor Analysis Consortium (CPTAC), "Radiology data from the clinical proteomic tumor analysis consortium lung squamous cell carcinoma [cptac-lscc] collection [data set]," 2018.

[48] J. Bijlsma and J. Wesseling, "Thematic collection: Check (cohort hip & cohort knee)," 2015.

[49] M. Weiner, P. Aisen, and R. Petersen, "Alzheimer's disease neuroimaging initiative."

[50] R. J. Marinelli, K. Montgomery, C. Long Liu, N. H. Shah, W. Prapong, M. Nitzberg, Z. K. Zachariah, G. J. Sherlock, Y. Natkunam, R. B. West, M. van de Rijn, P. O. Brown, and C. A. Ball, "Stanford tissue microarray database," *Nucleic Acids Res*, 2007.

[51] D. W. Scott, *Multivariate density estimation: theory, practice, and visualization.* John Wiley & Sons, 2015.

[52] S. Kullback, *Information theory and statistics.* Courier Corporation, 1959.

[53] R. H. Byrd, P. Lu, J. Nocedal, and C. Zhu, "A limited memory algorithm for bound constrained optimization," *SIAM Journal on scientific computing*, vol. 16, no. 5, pp. 1190–1208, 1995.

[54] K. Clark, B. Vendt, K. Smith, J. Freymann, J. Kirby, P. Koppel, S. Moore, S. Phillips, D. Maffitt, M. Pringle, L. Tarbox, and F. Prior, "The cancer imaging archive (tcia): Maintaining and operating a public information repository," *Journal of Digital Imaging*, vol. 26, no. 6, pp. 1045–1057, 2013.

[55] L. van der Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of Machine Learning*, vol. 9, pp. 2579–2605, 2008.

A Additional Figures and Tables

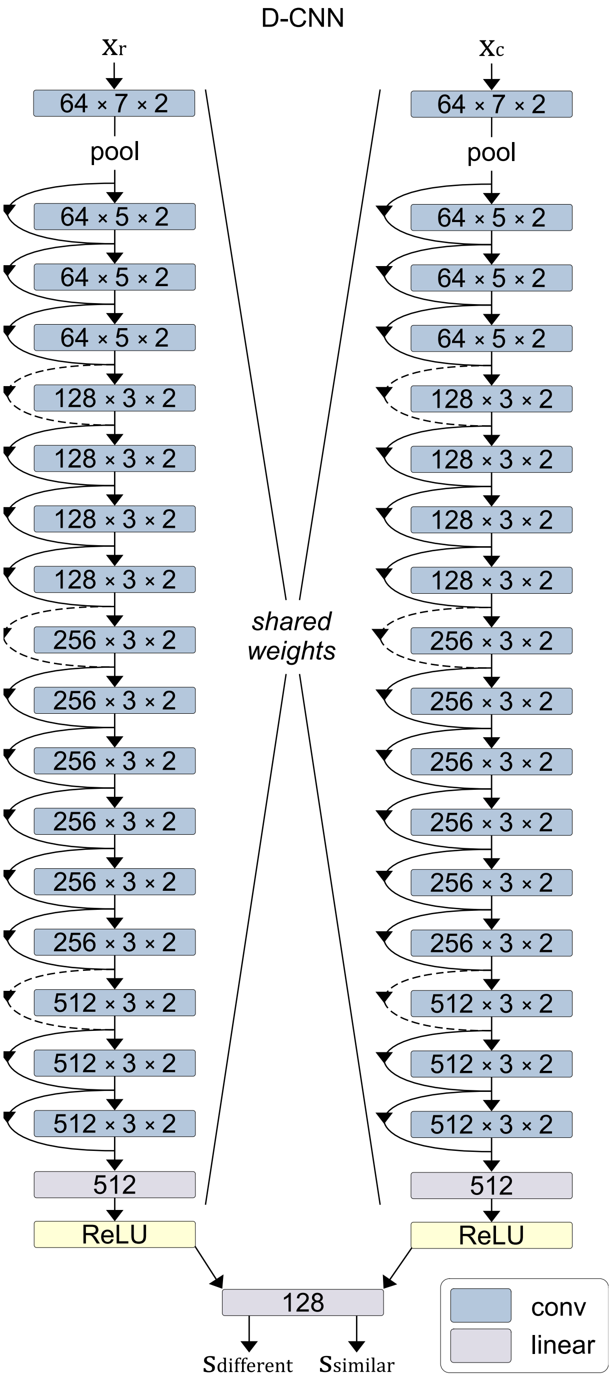

Figure A.1: The D-CNN model architecture, including all ResNet34 layers.

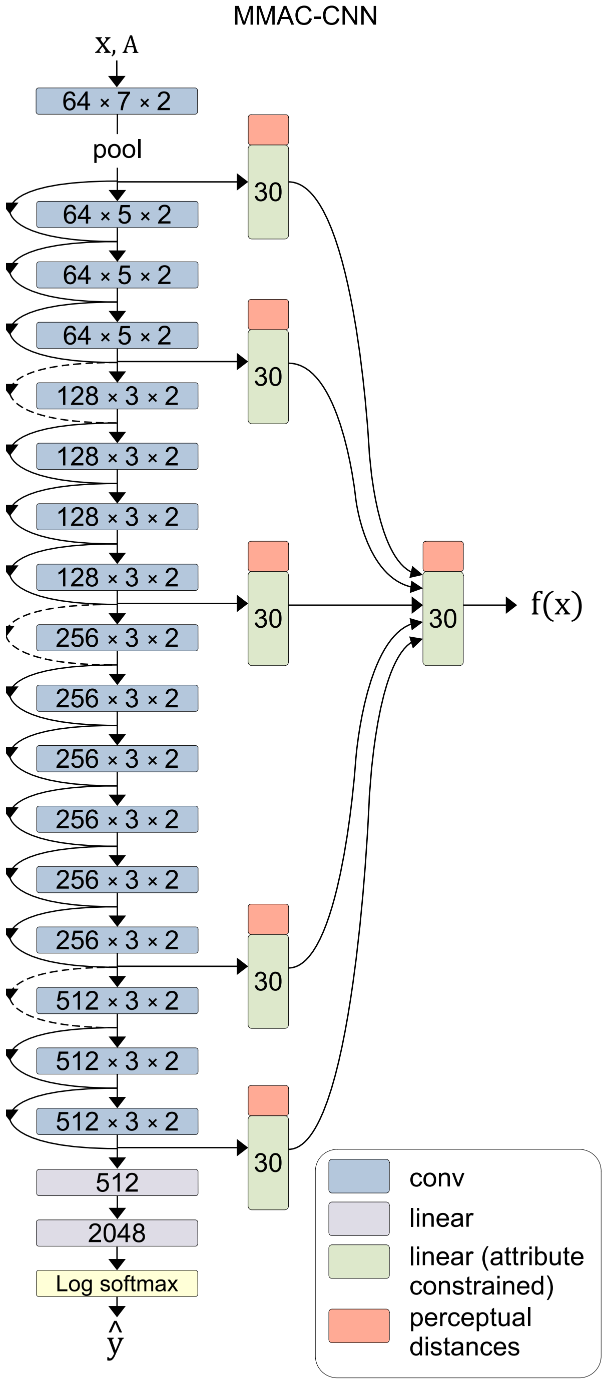

Figure A.2: The MMAC-CNN model architecture, including all ResNet34 layers.

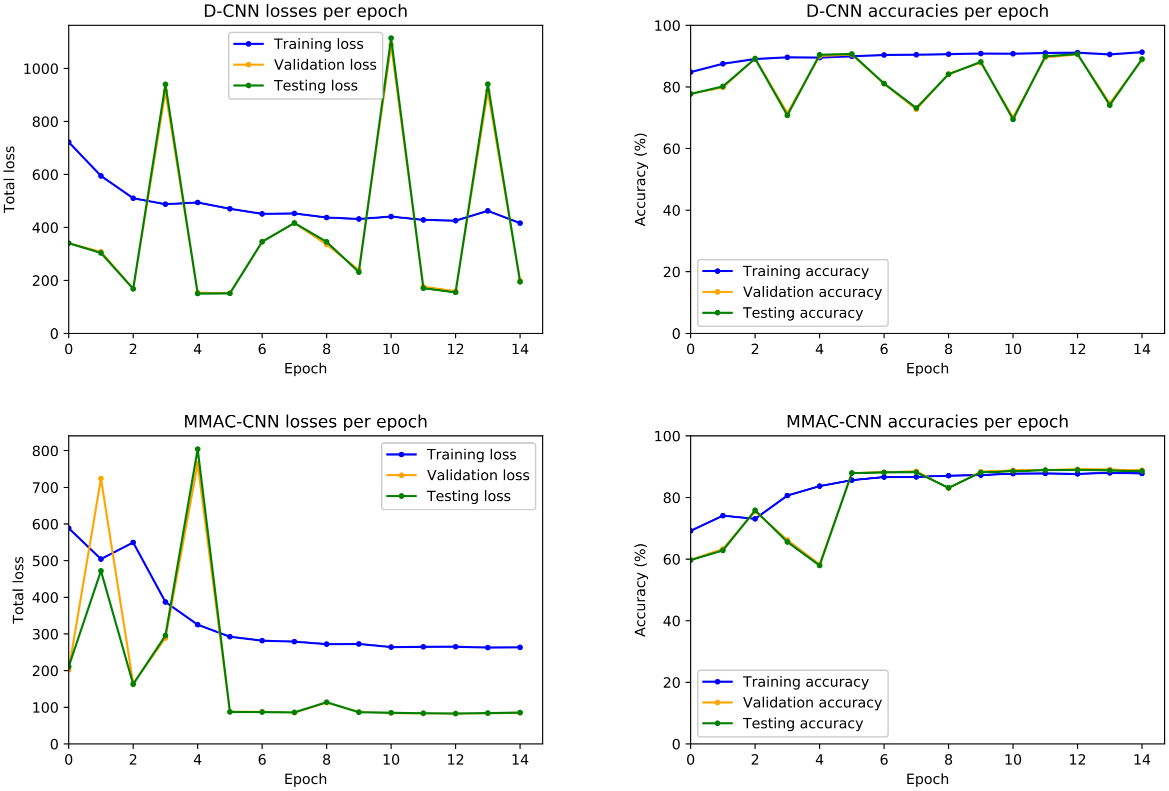

Figure A.3: D-CNN and MAC-CNN loss and accuracies per epoch, for one randomly-initialized pair of models.

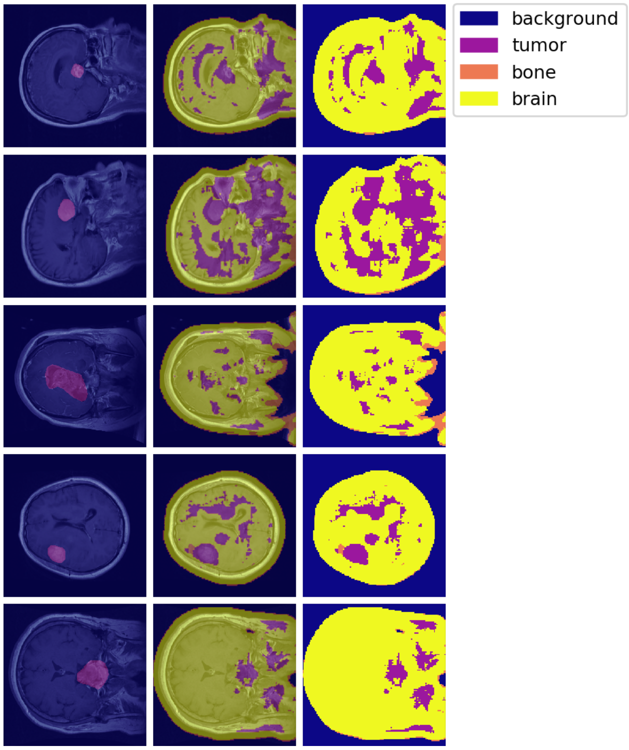

Figure A.4: Additional samples from evaluating MMAC-CNN category predictions on brain MRIs.

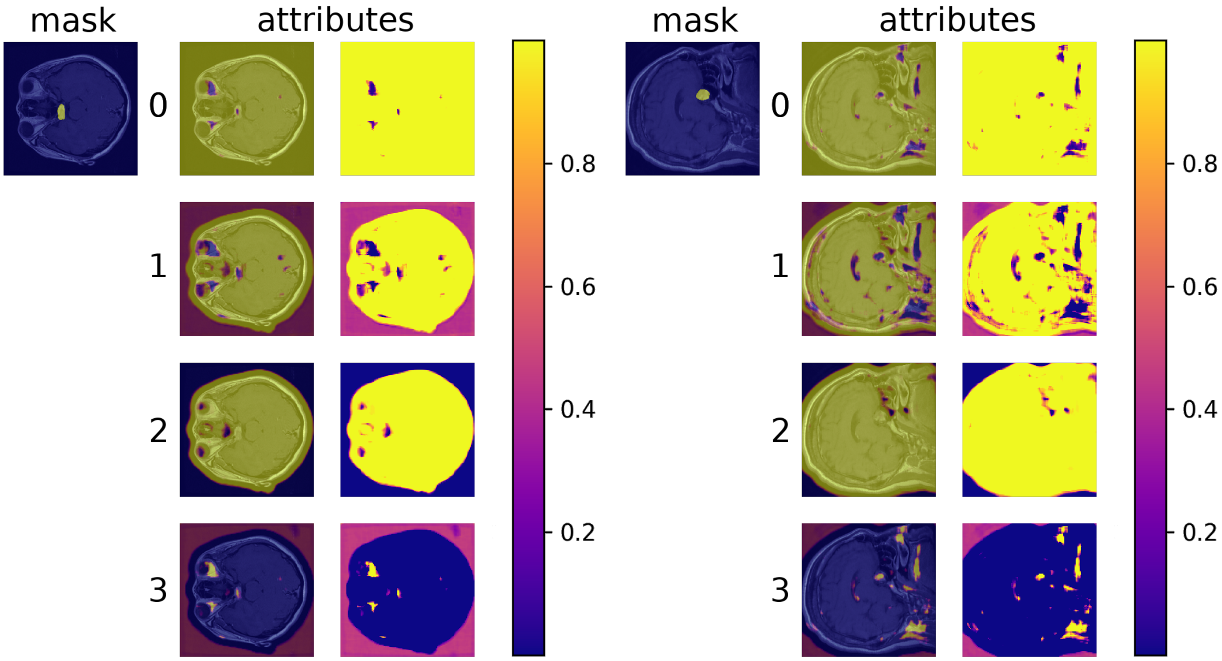

Figure A.5: Additional samples from evaluating MMAC-CNN attribute predictions on brain MRIs.

B Funding Sources

This research was funded by a University of Arkansas Honors College Team Grant.

C Code and Data Availability

The code used to implement and evaluate our method is available on GitHub:

- Source code: `https://github.com/cmolder/medical-materials`

The datasets used to evaluate our method are available at the following links:

- Cohort Knee and Cohort Hip (CHECK): `https://easy.dans.knaw.nl/ui/datasets/id/easy-dataset:62955`
- brain tumor dataset: `https://figshare.com/articles/brain_tumor_dataset/1512427`
- Brain-Tumor-Progression: `https://wiki.cancerimagingarchive.net/display/Public/Brain-Tumor-Progression`