CLASSIFICATION OF BRAIN TUMOR TYPES USING MRI IMAGES

STAT 6500 GROUP PROJECT REPORT

Cameron Conte, Yoon Ji Kim, Blue Webb, Noel Weidman and Zhizhen Zhao

1 Introduction : Brain Tumor Data Set

With advances in technology, image data such as MRI images have become prevalent in medical research. While MRI images are used to assess the existence of tumor, patients have to go through a biopsy or a surgery to determine the type of tumor. We study a MRI image data set to examine statistical classification methods. The data set is provided by a biomedical engineer, Jun Cheng, through the figshare website [3]. He published a journal publication on tumor classification through region augmentation using the same data set [5]. The images in the data set are 3064 two-dimensional brain MRIs with different sizes. They are from 233 patients who are diagnosed with one of three types of brain tumors; meningioma, glioma, and pituitary tumor. If there are multiple images from the same patient, they are different slices in the MRI. The total number of MRIs of meningioma, glioma, and pituitary tumor are 708, 1426, and 930 respectively.

The images are in gray-scale and are weighted to enhance their contrasts. While the original images are in different sizes, we have recaled them to 225×225 pixels through an R package called 'imager' using the nearest-neighbor interpolation method. These rescaled images are the inputs or a predictor variable in our classification models. The response variable is the tumor types; meningioma, glioma, and pituitary tumor. The data set also provides two other data types: tumor border data and tumor mask data. The tumor border data is the tumor shapes of each images which are obtained through an advanced computation method from the author. The borders are represented by a discrete points of coordinates. The tumor mask data are binary images that distinguishes the tumor shape and thus is bijective with the tumor border data. They are both derived from the gray-scaled images mentioned earlier. Although there are three different types of data per MRI slices, the contrast-inhanced gray-scale images will be studied mainly for classification.

2 Questions of Interest

Our goal is to compare the performance of classification methods in distinguishing tumor type using MRI image data as predictors. We compare Generalized Linear Models, Support Vector Machines, and a Convolutional Neural Network approach. In addition to comparing the three methods, we will compare the performance to the classifier introduced by Cheng et al [5].

We will investigate how training these classifiers on the whole images rather than an augmented tumor region as in [5] affects the overall performance of the models.

Analysis of tumor shapes or borders are not trivial since they are in different scales and the general statistical techniques are affected by rotation and warping in the input domain. Thus, we aim to align the tumor shapes and compute the mean shape of each tumor type using the statistical shape analysis. While a classification tool has not been developed for closed planar shapes, registering the shapes and deriving the mean shape would visualize the difference between the three classes of tumors.

3 Methodology

We split the data into training data of size 2451 (80%) and test data of size 613 (20%). We train the model and perform validation on the training set of images and assess the prediction error rate and sensitivity on the testing set. Generalized Linear Model (GLM), Support Vector Machine (SVM), Convolutional Neural Network (CNN) are built for image classification, and statistical shape analysis is adopted to study the tumor border data.

3.1 Generalized Linear Model with PCA

The generalized linear model allows us to link our predictor variables to the expected value of our response through a specified function. This is advantageous over the linear model as our vectorized image data is unlikely to follow a Gaussian distribution. Specifically, due to the high dimension of our data, we will perform principal component analysis and the subsequent PCA scores will be used as predictors in the model.

In order to determine the number of principal components we will utilize, we will examine the proportion of variability explained by each principal component and aim for a total proportion of 85 percent of variability explained by the components. This analysis will be performed using the ‘pca‘ package in R. By having our images stored as vectors of pixel intensities, we are able to directly apply PCA.

While aiming to explain approximately 85% of the variability using the principal components, we determined the final number of components in the GLM by fitting multiple models on the training set with varying numbers of principal components and examining the training accuracy. Models were fit using the ‘multinom’ function in the ‘nnet’ package in R. Figure 1 shows the training accuracy for various models and the red vertical line corresponds to the model that included 189 components which are needed to explain 85% of the variability. The model with 189 components offers a good trade-off of accuracy and reducing over-fitting and is ultimately the model that we chose.

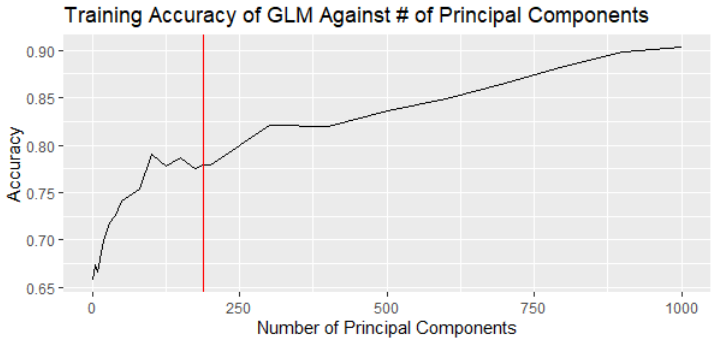

Figure 1: Training accuracy for GLM.

The test accuracy of the chosen GLM model was 77.16% and the confusion matrix (with true classes shown along the columns, and predicted classes shown along the rows) is shown in Table 1. Although the results are underwhelming, it is important to note that this is intended to be a baseline model to compare the more involved SVM and CNN models against.

	Meningioma	**Glioma**	**Pituitary**
Meningioma	84	42	5
Glioma	29	195	5
Pituitary	33	26	194

Table 1: Confusion matrix for GLM classifier

3.2 Support Vector Machines

We utilize support vector machines, a type of supervised learning model optimal for classification and regression applications. We hoped it would offer a compromise in terms of generalizability between our baseline method of GLM and the more advanced CNN. SVMs rely on having known class labels, which we have here as our tumor types: meningioma, glioma, and pituitary, which are classified as 1, 2, and 3 respectively. These known class labels will allow us to asses both training and testing error of our SVM model.

Because our image data is not linear, SVMs will work to find the optimal dividing hyperplane by using kernels to map our image data to a higher dimensional space. The principal component analysis conducted prior to use of GLM will be used here as well. This dimension reduction is necessary for the computation and tuning of SVM.

We visualize the first two principal components coded by their class labels to assess separability in Figure 2.

Overall, we are able to account for 85 percent of total variability by taking the first 189 principal components. While this is objectively still high dimensional, it entails a significant dimension reduction that will expedite classification with our support vector machine. Due to the data not being easily linearly separable, we opt for a radial kernel for our classifier. Table

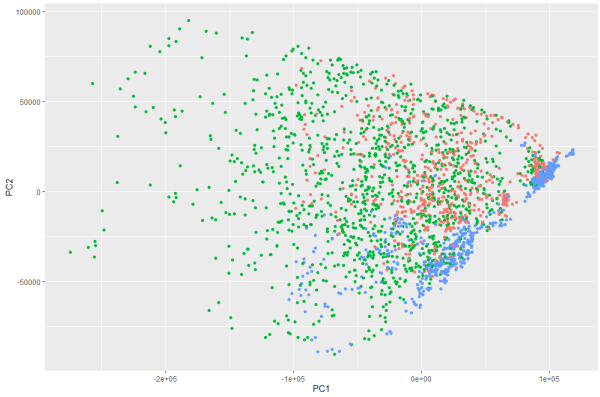

Figure 2: First two principal component scores.

	Meningioma	**Glioma**	**Pituitary**
Meningioma	114	16	0
Glioma	28	243	4
Pituitary	4	4	200

Table 2: Confusion matrix for radial SVM classifier

2 shows the confusion matrix (with true classes shown along the columns, and predicted classes shown along the rows) for this classifier, with the diagonals representing correctly classified tumor types.

We may also visualize the classification boundaries in a lower dimensional space, utilizing the model attained from the first 189 principal components to create boundaries based on the first two principal components. See Figure 3.

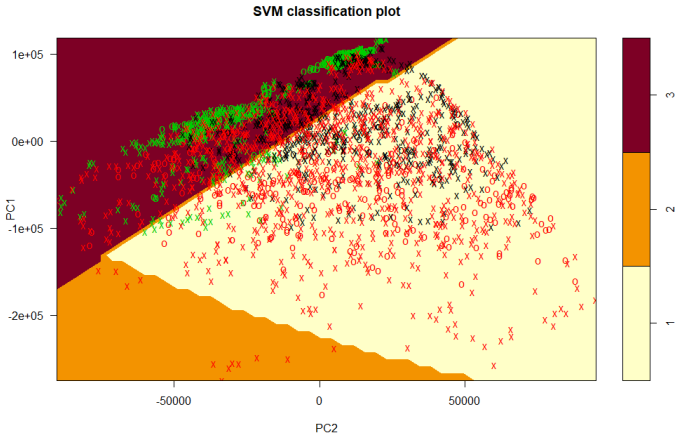

Figure 3: SVM boundary on first two principal components.

3.3 Convolutional Neural Network

Convolutional neural network (CNN) is a type of artificial neural network and is well-suited to extracting the higher level representations of images [9]. Compared with other classifiers, preprocessing the data is not as essential for CNN. A CNN accepts as input the raw pixel data of images in matrix form(a $225 \times 225 \times 1$ matrix in our model). The data is passed through a series of convolutional, nonlinear, pooling, and fully connected layers which process the data and generate the output. Figure 4 illustrates the end-to-end structure of a CNN [6]. It contains two convolution modules for feature extraction, and two fully connected layers for classification.

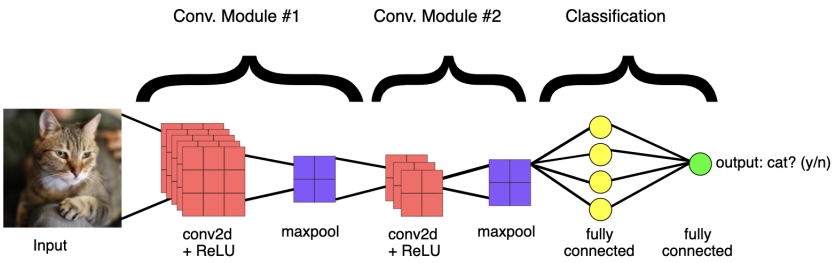

Figure 4: Example of the end-to-end structure of a CNN.

Each layer of the CNN plays a distinct role in processing the data before passing the information to the next layer. First, the input matrix is entered into the convolution layer. The convolution layer combines a linear convolution and a nonlinear activation function[9]. As filters slide over the input array, values of filters are multiplied by the original pixel values and all multiplications are summed up, resulting in a smaller matrix. Following the convolution, an activation function is added to bring nonlinearity into the model. If a previous convolution operation has identified some features, a detailed image is not necessary for further analysis, and so the image can be compressed to a less detailed image[9]. Thus, the pooling layer is included to perform downsampling on the size of image to save the processing time while preserving the most critical feature information[9]. Max pooling is used in our model. Additional layers may be added, such as batch normalization layers and dropout layers, which are used in our model. They could help prevent overfitting and allow faster training[9]. A fully connected layer is attached at the end of the model architecture to perform classification based on the features extracted by the convolutions. This layer contains a softmax activation function (used for multiclass classification) to output a n-dimensional vector, which is the predicted probability value for each of the n class label.

We developed our CNN architecture by referencing the results of [1], although ultimately our model architecture ended up being somewhat simpler. The entire model architecture is depicted in Figure 5, and exact parameters are in Table 3. While training our network, we split our training set into a sub-training and validation set, with this smaller validation set comprising 20% of the initial training data. The final results were then computed for the held out test set. Our model architecture consisted of several stacked units, each consisting of convolutional layers (each of which in turn used the rectified linear

unit activation function), followed by a max pooling layer and a dropout layer. We used a larger kernel size of 5×5 in the first layer of our model to help it learn larger spatial features. Then we decreased the kernel size as the depth of the model increased. The max pooling layer could give an output two times smaller than the input. We utilized batch normalization between layers. We used dropout layers as well, with a moderate rate of 0.5. Throughout the model, we increased the number of filters in each convolutional layer. In total, our model contained 75,788,643 trainable parameters.

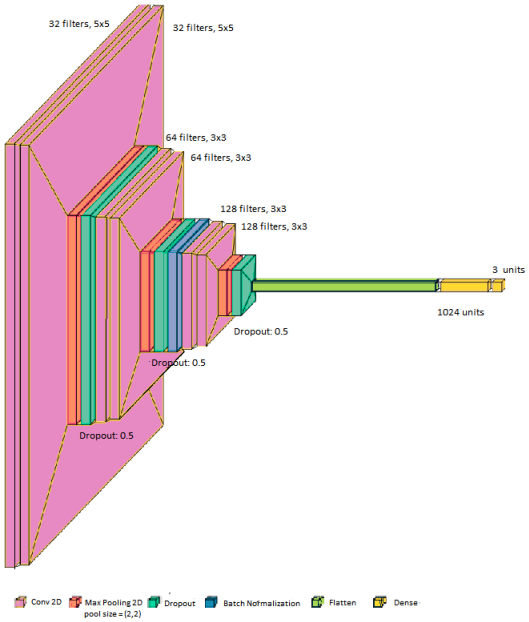

Figure 5: Final CNN architecture

Figure 6: Training and validation results

Table 3: Description of CNN layers

Layer	Parameters
1	Conv2D(filters=32, kernel_size = (5,5), "relu")
2	Conv2D(filters=32, kernel_size = (3,3), "relu")
3	MaxPooling2D(pool_size=(2,2))
4	Dropout(rate=0.5)
5	Conv2D(filters=64, kernel_size = (3,3), "relu")
6	Conv2D(filters=64, kernel_size = (3,3), "relu")
7	MaxPooling2D(pool_size=(2,2)
8	Dropout(rate=0.5)
9	BatchNormalization
10	Conv2D(filters=128, kernel_size = (3,3), "relu")
11	Conv2D(filters=128, kernel_size = (3,3), "relu")
12	MaxPooling2D(pool_size=(2,2)
13	Dropout(rate=0.5)
14	Flatten
15	Dense(units=1024, "relu")
16	Dense(units=3, "softmax)

	Meningioma	**Glioma**	**Pituitary**
Meningioma	138	11	2
Glioma	7	252	0
Pituitary	1	0	202

Table 4: Confusion matrix for CNN

	Precision	**Recall**	**F1-Score**
Meningioma	0.91	0.95	0.93
Glioma	0.97	0.96	0.97
Pituitary	1.0	0.99	0.99

Table 5: Other Metrics for CNN

The model was developed in the `keras` library in Python. We used the Adam optimizer with a learning rate of 0.0001. Our batch size was 16, and the model was trained for 45 epochs. These hyperparameters were tuned based on performance of the classifier on the validation set we formed from the training data. Figure 6 shows the accuracy and loss for our training and validation set over the training epochs. The accuracy of our final model on the validation set was 96.74%. The final model performed very well on the test set, with an accuracy of 96.57%. These results are similar to those found by [1], whose (more robust) estimate of the accuracy was calculated using 10-fold cross validation, and found to be 96.56%.

The confusion matrix (with predicted classes shown along the rows, and true classes shown along the columns) shows that the CNN most commonly misclassified gliomas as meningiomas, and vice versa. This makes sense, as pituitary tumors tend to be more localized and their MRIs are more distinctive.

3.4 Statistical Shape Analysis

Two dimensional shapes of tumors have to be analyzed considering their rotations and warping in their input domains. Shapes that are rotated and warped from the mean shape makes it difficult to derive the sample mean shape and align the observed shapes. Statistical shape analysis introduces square-root velocity function (SRVF), $q(t)$, as below and derives statistical

inference that is invariant to rotations and warping in the input domain through the Fisher-Rao Riemannian metric.

$$q(t) = \frac{\dot{f}(t)}{\sqrt{\dot{f}(t)}}, \tag{1}$$

where $f(t)$ represents a planar curve, $\dot{f}$ is the gradient of function f and t is the input such as coordinates. When two planar shapes which are indicated by SRVFs, q_1 and q_2, and γ is a phase or warping function where (q, γ) is a input reparametrization of q using γ. We can easily prove that the SRVF representation makes the $\mathbb{L}^2$ distance between two shapes invariant to any warping; $\|q_1 - q_2\| = \|(q_1, \gamma) - (q_1, \gamma)\|$. Thus, the shape distance can be derived through $\|q_1 - q_2\|$. Via the shape distance computation, we may obtain the mean shape, called Karcher mean, as in figure 7.

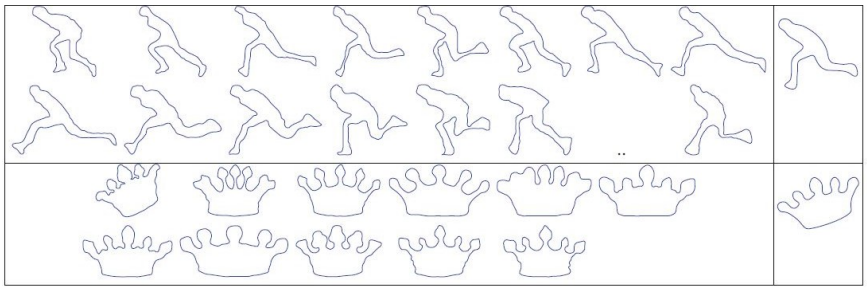

Figure 7: The shapes in each row illustrates the derivation of the mean shape from the observed shapes. In the left boxes are the observed shapes and the corresponding mean shape is plotted in the right box in the same row.

4 Results

4.1 Classification

After performing principal component analysis on the training image data, we used these features as inputs for GLM and SVM classifiers. The GLM had a higher training error of 22.84%. We tuned the support vector classifier using a radial kernel to reduce our testing error. It was found that γ = 0.009 produced the optimal classifier with a testing error of 9.1%. Our CNN proved to be the most successful classifier with a testing error of 3.4%. Since this data set has been used in multiple studies, we can compare these results to some of them. Cheng et al. were interested in using the augmented tumor region as the region of interest for their classifiers; they also examined the results of three feature extraction techniques (intensity histogram, gray level co-occurrence matrix (GLCM), and bag-of-words (BoW) model)[4]. Using the the BoW technique with the augmented tumor region (their paper details the specifics of their methodology), they evaluated the performance of several different classifiers: SVM, sparse-representation classifier (SRC), and k-NN for several values of *k* [4]. Their results are shown in Table 6.

Notably, they found that SVM achieved the highest accuracy. The accuracy of their SVM classifier was close to ours, where we used the principal components as features. Our CNN achieved results comparable to those of Badža and Barjaktarović,

Classifier	SVM	SRC	1-NN	3-NN	7-NN	15-NN	45-NN
Accuracy	91.14	86.55	80.01	81.69	83.14	83.37	83.09

Table 6: Results of Cheng et al. [4]

whose CNN achieved an accuracy of 96.56% (computed using ten-fold cross-validation) [1]. To build more effective classifiers in the future, we might consider adopting some of the preprocessing techniques outlined in [4] and [1] and evaluating our models using k-fold cross-validation.

Classifier	**Test Error Rate**	**Sensitivity**
GLM	**22.84%**	**77.16%**
SVM	**9.1%**	**90.86%**
CNN	**3.4%**	**96.57%**

Table 7: Summary of Classification Result.

4.2 Karcher Mean Shapes of Tumors

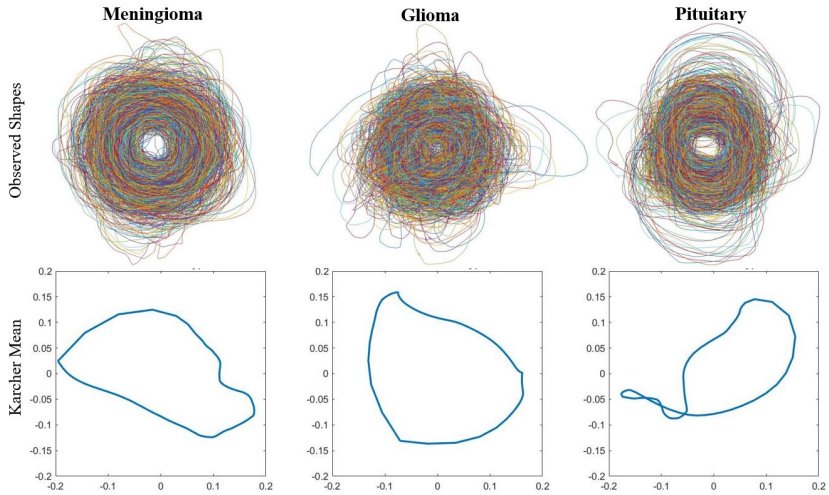

Figure 8: The plots in the first row are the observed samples in each class and the shapes in the second row are the mean shapes of corresponding classes.

We implement the statistical shape analysis to derive the mean shape of meningioma, glioma, and pituitary tumor and examine the differences between their mean shapes visually. Figure 8 is a visualization of the observed shapes that are rescaled and their mean shapes and Table 8 is the shape distance between the mean shapes of three tumor types. Comparing the Karcher means of meningioma and glioma, the mean shape of glioma is more round while the mean shape of meningioma is more elliptical. We can observe that the mean shape of pituitary is distorted which may be due to its constrained area in the brain or very different shaped observations. While the shape distance between the three Karcher means are similar to each other, the

distance between meningioma and glioma is greater than two other comparisions; meningioma vs. pituitary and glioma vs. pituitary.

Tumor Types	Shape Distance
Meningioma & Glioma	12.1316
Meningioma & Pituitary	10.9571
Glioma & Pituitary	10.8053

Table 8: Shape distance between mean shapes of three tumor types.

5 Conclusion

We performed principal component analysis in order to reduce dimensionality. We used the principal components as features in a GLM, achieving a testing error of 22.84%, and a support vector classifier with a radial kernel, achieving a testing error of 9.1%. For the SVM, this was attained for a small value of the parameter γ (0.009), which incurs low variance at the expense of high bias. We also built a convolutional neural network. After fine-tuning the architecture, we found this to be the most effective of the three classifiers, with a test error of 3.4%. The performance of our CNN classifier found out to be comparable to the classifiers in the published literature. We also visualized the mean shapes of meningioma, glioma, and pituitary and derived the pairwise shape distance between the mean shapes. The distance between meningioma and glioma tumor types was greater than the other two distances.

All data set and code are available at MRI-code-Data.

References

[1] Badža, M., and Barjaktarović, M. (2020). Classification of Brain Tumors from MRI Images Using a Convolutional Neural Network. Applied Science, 10(6), 1999. https://doi.org/10.3390/app10061999.

[2] Peter N. Belhumeur, Joao P. Hespanha, and David J. Kriegman. (1997). Eigenfaces vs. Fisherfaces: Recognition Using Class Specific Linear Projection, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *19*(7),711–720.

[3] Jun Cheng. (2017). Brain Tumor Dataset, *Figshare*, https://doi.org/10.6084/m9.figshare.1512427.v5.

[4] Jun Cheng. (2015). Enhanced Performance of Brain Tumor Classification via Tumor Region Augmentation and Partition, *PloS one 10.10*.

[5] Jun Cheng. (2016). Retrieval of Brain Tumors by Adaptive Spatial Pooling and Fisher Vector Representation, *PloS one 11.6*.

[6] Ml practicum: Image classification | machine learning practica. (n.d.). Retrieved April 01, 2021, from https://developers.google.com/machine-learning/practica/image-classification/convolutional-neural-networks

[7] Anuj Srivastava and Eric P. Klassen. (2016). Functional and Shape Data Analysis *Springer-Verlag New York*.

[8] Phillip Wagner. (2012). *Face Recognition with Python*. Bytefish. https://www.bytefish.de/pdf/facerec_python.pdf

[9] Yamashita, R., Nishio, M., Do, R. K. G. and Togashi, K. (2018). Convolutional neural networks: an overview and application in radiology. Insights into imaging, 9(4), 611-629.