Traditional Learning Models Vs. Neural Networks in Detecting Brain Hemorrhages

Andrew Pham, Khang Huynh, Sebastian Duran, Stephanie Snow

Abstract

This study proposes several types of traditional learning models as well as neural networks used to predict if a patient has a brain hemorrhage or not. All of these models utilized the same dataset of a little under 7000 grayscale images at a resolution of 512 x 512. The validation set was tested across all models, yielding measurable data to inform a comprehensive comparison. The two traditional learning models used were Support Vector Machine (SVM) and Logistic Regression while three neural networks were used. Those three include Convolution Neural network (CNN), Recurrent Neural Network (RNN), and Attention RNN. It was found that CNN performed the best out of all five models with a validation accuracy of 73.1%. Surprisingly, the other neural networks performed similarly to the traditional learning models where the average validation accuracy of those models were 57.3%. We therefore find CNN as the optimal model for predicting brain hemorrhages as it can pick up complex patterns from images better than traditional learning models and the other neural networks.

1. Introduction

The human brain serves as the command center for the entire body, controlling how we move, respond, and think. A brain hemorrhage, otherwise known as a brain bleed, is when a blood vessel leaks or bursts causing blood to pool within the skull and brain [1]. Brain hemorrhages are a serious condition that could be fatal if left untreated. It is estimated that there are approximately 40,000 to 67,000 brain hemorrhages every year in the United States. From these cases, the mortality rate ranges from 35% to 52% within 30 days after being admitted to a hospital. It is estimated that about half of the deaths occur within the first 24 hours. Of the survivors, only 20% are expected to be independent after six months [2]. Due to the rapid and often fatal nature of this disease, it is of utmost importance for patients to receive appropriate medical treatment as early as possible.

Consequently, the tools in accurately identifying a brain hemorrhage is of utmost importance. Medical imaging plays a critical role in diagnosing neurological conditions and is essential in understanding the severity and location of a brain hemorrhage. MRI and CT scans are analyzed to detect various types of brain hemorrhages and potential underlying causes. Computed Tomography (CT) scans which are 3D representations of the body created using a combination of X-rays and computer processing to generate detailed cross-sectional images. These scans are widely used in medicine because they offer fast and non-invasive visualization of organs, bones, and tissue with great accuracy. Because of this, they are the most common method used to detect brain hemorrhages and even show possible related injuries such as skull fractures or brain swelling [3]. However, there are challenges to manually interpreting CT scans which can lead to misdiagnosing or even failing to see hemorrhages which risk the patient's life. Being able to build on existing diagnostic infrastructure by using Machine Learning tools can enhance diagnostic accuracy and can be crucial in saving time and potentially a patient's life and wellbeing.

The purpose of this project is to provide advancements in detecting brain hemorrhages by using machine learning models, and in doing so, improve early diagnosis and patient recovery outcomes. To achieve this, the project evaluates the performance of neural network models compared to traditional learning models by using a Kaggle dataset containing CT images of brains with and without hemorrhages. The dataset used for this project consists of 6,795 CT images that are sized 512 x 512 pixels each. This project uses two traditional machine learning models, Support Vector Machine (SVM) and Logistic Regression, as well as three neural network models, Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), and Attention-RNN. To implement the project, SVM and Logistic Regression are used as baseline models to compare accuracy scores with those from deep learning approaches. Then, the focus shifts to CNN and RNN with their ability to detect complex patterns. Finally, discrepancies in accuracy and computational time are evaluated across the different models to determine the most effective approach.

2. Related Work

The application of machine learning in medical imaging has been widely researched, offering valuable insights for this study. For example, a 2019 study published in Medical Image Analysis evaluated the performance of CNNs versus traditional models like SVM in brain tumor classification. The findings demonstrated a clear advantage for CNNs, with accuracy improvements of over 10 percentage points. This aligns with our results, where CNNs significantly outperformed traditional methods such as SVM and Logistic Regression. [4] [5]
In a related study presented at the international Conference on Computational Intelligence and Applications in 2020, researchers explored the effectiveness of attention based RNNs for medical image analysis. While these models provided enhanced interpretability by focusing on key regions of interest, they faced challenges with overfitting when applied to smaller datasets. Similar limitations were observed in our work with Attention RNNs, where overfitting impacted validation accuracy. [5] [6]
Additionally, a 2018 article in Frontiers in Neuroscience reported CNN test accuracies exceeding 95% in brain hemorrhage detection. However, these results were achieved on a dataset comparable in size to ours, raising concerns about potential overfitting or inadequate validation techniques. Drawing from this, we employed a separate validation set to mitigate such risks and ensure the robustness of our findings. [5]
By learning from these prior works, we implemented strategies to optimize our models while addressing common pitfalls like overfitting and data imbalance, ultimately achieving a balance between performance and generalizability.

3. Proposed Method

In this project, we implement five models: two traditional machine learning algorithms and three neural network architectures. The traditional models include Support Vector Machine (SVM) and Logistic Regression, chosen for their established effectiveness in binary classification tasks.
For the neural network approaches, we utilize a Convolutional Neural Network (CNN), a Recurrent Neural Network (RNN), and an Attention based Recurrent Neural Network (Attention RNN). The CNN is designed to extract spatial features for pattern recognition. The RNN is employed to explore sequence based learning, while Attention RNN incorporates attention mechanisms to focus on the most relevant features within the data.
All models are trained and validated on the same dataset of approximately 7,000 gray scale images, ensuring a consistent basis for comparison. Model accuracy, defined as the number of correct predictions divided by the total number of predictions, serves as the primary performance metric for evaluating these approaches.

3.1 Support Vector Machine (SVM)

Support Vector Machines (SVM) are traditional supervised machine learning algorithms that can be used for both classification and regression tasks. They are particularly effective for binary and multiclass classification. SVMs can classify non-linear datasets using kernels which map the data into higher-dimensional spaces where it becomes linearly separable. This project will use SVM's binary classification to determine whether or not there is a hemorrhage in the CT images. Additionally, the model will evaluate three kernels–linear, radial basis function (RBF), and polynomial–to determine which of them achieves the highest accuracy.

3.2 Logistic Regression

Logistic Regression is the other traditional machine learning algorithm that will be used for this project. As a supervised learning technique, it is primarily used for binary classification tasks, particularly in predicting whether or not an instance belongs in a given class. This makes it well-suited in predicting the presence or absence of hemorrhages in this dataset. However, unlike SVM, logistic regression assumes linearity and struggles with non-linear datasets. Given its simplicity and interpretability, logistic regression will serve as a benchmark model for this dataset, providing a basis for comparison against the more advanced models.

3.3 Convolutional Neural Network (CNN)

A Convolutional Neural Network is a deep learning model designed to process structured data like images by using convolutional layers to automatically detect patterns such as edges, shapes, and textures for tasks like classification. Instead of the basic neural network multiplying the input by the weight vector as it passes through layers, CNN applies convolution with a filter to the images to extract features known as feature maps. These feature maps are then pooled to reduce the spatial dimensions. Additional layers can also be provided like normalization and dropping out, but in the end it gets fed into the fully connected layer to make a prediction.

Type	Details	Input Size
Convolution	(32)3x3	(128, 128, 3)
Batch Norm.		(126, 126, 3)
Max Pooling	2x2	(126, 126, 3)
Dropout	30%	(63, 63, 32)
Convolution	(64) 3x3	(63, 63, 32)
Batch Norm.		(61, 61, 64)
Max Pooling	2x2	(61, 61, 64)
Dropout	30%	(30, 30, 64)
Convolution	(128) 3x3	(30, 30, 64)
Batch Norm.		(28, 28, 128)
Max Pooling	2x2	(28, 28, 128)
Dropout	30%	(14, 14, 128)
Convolution	(256) 3x3	(14, 14, 128)
Batch Norm.		(12, 12, 256)
Max Pooling	2x2	(12, 12, 256)
Dropout	30%	(6, 6, 256)
Convolution	(512) 3x3	(6, 6, 256)
Batch Norm.		(4, 4, 512)
Max Pooling	2x2	(4, 4, 512)
Global Avg. Pool		(2, 2, 512)
Dense	(256) Relu	512
Batch Norm.		256
Dropout	50%	256
Dense	(1) Sigmoid	256

3.4 Recurrent Neural Network (RNN)

A Recurrent Neural Network (RNN) is a deep learning model designed to process sequential data, such as time series or text, by utilizing recurrent connections to capture temporal dependencies and patterns. Unlike basic neural networks that process inputs independently, RNNs maintain a hidden state that carries information from previous time steps, allowing them to model the sequence's context. At each time step, the input is combined with the hidden state through a set of learnable weights, enabling the network to process the sequence element by element. Variants like Long Short-Term Memory (LSTM) introduce mechanisms to address issues like vanishing gradients, improving their ability to learn long-term dependencies. Finally, the RNN outputs predictions for tasks like classification, prediction, or sequence generation based on the processed sequential data.

Type	Details	Input Size

Reshape	Timesteps = 128
	Features = 128 * 3
		(128, 128, 3)

LSTM	(128) ReLU	(128, 384)
Dropout	30%	(128, 128)
LSTM	(64) ReLu	(128, 128)
Dropout	30%	(64)
Dense	(64) ReLu	(64)
Dropout	50%	(64)

Dense	(1) Sigmoid	(64)

3.5 Attention RNN

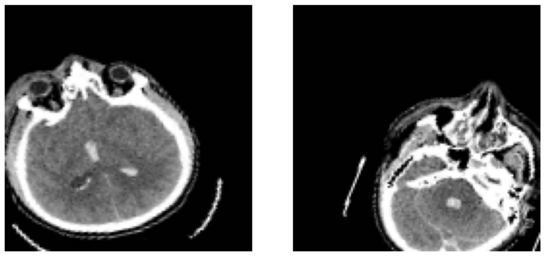
An attention-based RNN is a variant of a traditional RNN that incorporates an attention mechanism to dynamically assign importance to specific time steps in the input sequence. After the feedforward computation, the attention mechanism calculates attention weights for each time step's hidden state, based on its relevance to the final prediction [7]. These attention weights enable the model to focus on the most informative parts of the sequence, given all time steps, rather than treating all time steps equally. This approach provides a more nuanced and holistic way of determining which hidden states contribute most significantly to the final output, enhancing the model's ability to capture long-range dependencies and subtle temporal patterns. [7] [8] It is important to note that RNNs are primarily designed for processing sequential data, such as time series or natural language, where capturing temporal dependencies is critical. [9] They are not inherently tailored for image classification tasks, which typically involve spatial patterns in data. Image classification models often leverage Convolutional Neural Networks (CNNs), which are specifically designed to extract spatial features. [10] However, RNNs with attention mechanisms have been explored for certain specialized image tasks, such as image captioning or sequence prediction, where sequential dependencies are also important. [11]

Type	Details	Input Size
LSTM	128 units, tanh, L2 regularizer(0.01)	(128, 128)
dropout	30% dropout rate	(128, 128)
LSTM	64 units, tanh, L2 regularizer(0.01)	(128,64)
Dropout	30% dropout rate	(128, 64)
Attention mechanism	Standard attention layer	(128,)
Dense	128 units, relu, L2 regularizer(0.01)	(128)
Dropout	30% dropout rate	(128)
Dense(output)	1 unit, sigmoid, L2	(1)

4. Experiments

4.1 Dataset

The Kaggle dataset used contains 6,795 grayscale CT images from Near East University Hospital patients. The images were stored as JPGs and were in subfolders for every patient as multiple pictures were taken rotated around the head. The dataset was not balanced as 4,105 of the images were labeled hemorrhagic while 2,689 were labeled normal. The images were preprocessed by applying random horizontal and vertical shifts, rotations, zooms, shearing, and horizontal flipping. The Attention RNN model resized the images to 244 x 244 while every other model resized to 128 x 128.



Preprocessed Images

4.2 Support Vector Machine (SVM)

The SVM model was trained using the full Kaggle dataset, which was split with 70% training data (4,757 images) and 30% validation data (2,031 images). Before training, each image was reshaped into 2D data for compatibility with the SVM model. Features and labels are extracted from the dataset using *extract_features* from sklearn and the data was standardized using *StandardScaler* to improve the model's performance and reduce sensitivity to feature scales. The model was trained using *SVC* with an RBF kernel, and predictions were made on the validation set to evaluate accuracy.

To further explore performance, the model was also trained using two other kernels, linear and polynomial, to determine which achieved the higher accuracy. SVM hyperparameter tuning for C and gamma was attempted on this model using *GridSearchCV*; however, the computational time proved prohibitive, with the process taking over five hours without completion on the full dataset. Attempts to reduce computational time by limiting the dataset size or the parameter search range resulted in missing hemorrhagic predicted data. Consequently, the model's analysis focused on comparing kernel types without the extensive hyperparameter optimization.

4.3 Logistic Regression

The Logistic Regression model was trained using the full dataset, which was split with 70% training data (4,757 images) and 30% validation data (2,031 images). Before training, each image was reshaped from a 3D array (height x width x channels) to a 1D array to make it compatible with logistic regression. Given that each image was 128 x 128 pixels with 3 color channels, each 1D array would have 128 x 128 x 3 = 49,152 features. The model was then trained using Python's *LogisticRegression* from sklearn with a max iteration of 1000. After training, predictions were made on the validation set to evaluate the model's accuracy.

4.4 Convolutional Neural Network (CNN)

The CNN model was trained using 70% of the dataset and the other 30% as a validation set. The model was constructed using TensorFlow 2 and Keras, trained using Google Collab's T4 GPU, and the architecture of the CNN model can be seen in Section 3. In addition to the described model, every convolution layer used L2 regularization to combat overfitting. The learning rate was initially set to 0.001, but changed due to the learning rate scheduler by evaluating the validation loss on every epoch and adjusting as needed. The batch size was set to 32 to conserve memory and the model ran 100 epochs. After the model trained, the validation set was used again to evaluate the final score.

4.5 Recurrent Neural Network (RNN)

The RNN model was trained using 80% of the dataset for training and 20% for validation, utilizing TensorFlow 2 and Keras as the framework. The model was developed and trained on Google Colab's T4 GPU for efficient computation. Before training, the input images were reshaped into sequences with 128 timesteps and 384 features to accommodate the RNN architecture.

The model architecture included two stacked RNN layers, where each layer used ReLU activation functions, followed by dropout regularization to prevent overfitting. The fully connected dense layers provided further non-linear transformations, with the final output layer using a sigmoid activation for binary classification.

The training process applied data augmentation techniques, such as rotation, zoom, and flipping, to enrich the training dataset. The learning rate was initially set to 0.001 and was adjusted dynamically using a scheduler based on the validation loss. The batch size was fixed at 32 for memory efficiency, and the model was trained for 50 epochs. After training, the validation set was reused to evaluate the model's performance.

4.6 Attention RNN

The attention-based RNN model consisted of two primary components. The first component comprised two stacked LSTM layers, each utilizing a tanh activation function and a kernel regularizer with a value of .01 to prevent overfitting. Each LSTM layer was followed by a dropout layer with a dropout rate of 0.3. After the second LSTM layer, an attention mechanism was applied to focus on the most relevant features in the input sequence, enabling the model to prioritize key information for predictions. The second component included a dense layer with 128 neurons, a ReLU activation function, and a kernel regularizer of 0.01, followed by another dropout layer with a 0.3 dropout rate for additional regularization.

5. Results

The models demonstrated varying levels of performance, with convolutional neural networks (CNNs) outperforming others due to their superior ability to capture spatial patterns in image data. The CNN achieved the highest training accuracy (~94%) but exhibited significant overfitting, as evidenced by the gap between its training accuracy and validation accuracy (~73.1% after 50 epochs). Support Vector Machines (SVMs) performed moderately, with the RBF kernel achieving the highest accuracy (59%), followed closely by the polynomial kernel (58%) and the linear kernel (52%) logistic regression, used as a benchmark, achieved the lowest accuracy at 52%.

Recurrent Neural Networks (RNNs) with basic temporal processing capabilities achieved moderate performance, with a training accuracy of ~67.8% and a validation accuracy of ~58.9%, but struggled with generalization due to overfitting. Similarly, the attention-based RNN showed only marginal improvements over the standard RNN, with a training accuracy of 0.78 and a validation accuracy of 0.60, indicating difficulties in differentiating subtle differences between classes. Across models, overfitting remained a prevalent issue, particularly in the CNN and RNN architectures, despite the application of dropout, L2 regularization, and data augmentation techniques.

Future improvements could involve enhancing regularization, increasing dataset diversity, and exploring advanced architectures or preprocessing methods to address overfitting and improve generalization.

5.1 Support Vector Machine (SVM)

The kernel with the highest test accuracy score was RBF with 59%. The computational time it took to run the model was ~30 minutes. The polynomial kernel was close with an accuracy of 58% and the linear kernel had the lowest accuracy of the three at 52%. The table below shows the accuracy of each of the three kernels:

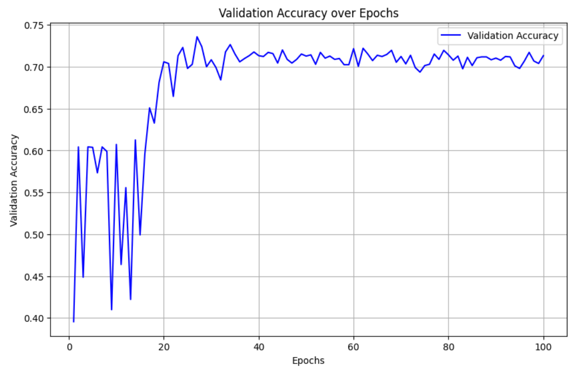
Kernel	Accuracy
RBF	59%
Polynomial	58%
Linear	52%

5.2 Logistic Regression

Logistic Regression was the benchmark for the other models and as expected had the lowest accuracy amongst them at 52%.

5.3 Convolutional Neural Network (CNN)

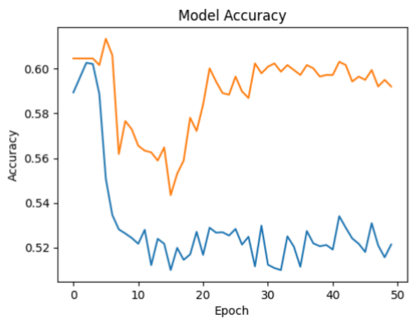
As anticipated, CNN performed the best due to its capability to capture intricate patterns from images. The CNN model with 5 layers achieved a training accuracy of ~94% while having a validation accuracy of 73.1% for 50 epochs taking ~35 minutes and 71.3% for 100 epochs taking ~1 hour which can be seen in the graph:



As can be seen, there was no point to continue training the model after ~25 epochs as the validation accuracy hovered around 71%. Since the model achieved a higher training accuracy than validation accuracy by approximately 20%, this indicates an overfitting problem. Even with random modifications to the original images during preprocessing, dropping out 30% after each layer, and applying L2 regularization in each convolution, this was not enough to stop the model from overfitting to the training data.

5.4 Recurrent Neural Network (RNN)

The RNN model exhibited moderate performance in handling sequential data by utilizing temporal patterns. The RNN model, comprising two layers with ReLU activation and dropout regularization, achieved a training accuracy of approximately 67.8% and a validation accuracy of around 58.9% after 50 epochs, as illustrated in the accuracy plot. The training loss was 0.5816, while the validation loss was 0.6973, indicating that the model struggled to generalize well to unseen data.

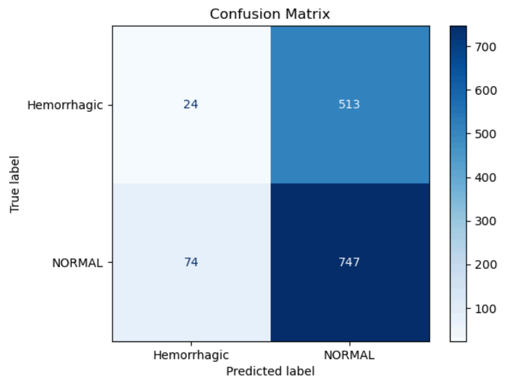


The accuracy plot indicates that training accuracy was consistently higher than validation accuracy, suggesting an overfitting problem. Despite applying 30% dropout after each layer, the model was unable to fully prevent overfitting. The confusion matrix reveals that the model correctly classified 807 hemorrhagic cases, but misclassified 531 normal cases, resulting in a significant false-positive rate.

Random modifications during preprocessing, such as reshaping and dropout, provided some resistance to overfitting, but these were insufficient to bridge the gap between training and validation performance. Future work could explore using advanced architectures like LSTM or GRU, increasing the dataset size, or applying more robust regularization techniques to improve generalization. Additionally, implementing a learning rate scheduler or increasing the diversity of the training data could further enhance performance.

5.5 Attention RNN

The model achieved a training accuracy of 0.78 with the training loss converging to approximately 0.67. The validation accuracy reached 0.60, and the validation loss similarly converged to 0.67. Despite the inclusion of an attention mechanism, the model only made marginal improvements, and based on the validation loss and accuracy, the model underperformed in general. In addition, based on the confusion matrix, the model overfitted the negative class which is the class that does not have a hemorrhage, so even though the loss for both the training and the validation were relatively low, the majority of the error was attributed to mostly one class. This issue stems from the model's inability to differentiate subtle differences between both classes.



5.6 Discussion

Traditional machine learning algorithms achieved less than 60% accuracy with Logistic Regression having the lowest accuracy of all the models at 52% and SVM being 59%. Among neural network models, RNN did unexpectedly poorly at 57%, even worse than SVM, while Attention-RNN only achieved a modest improvement at 60%. CNN had the highest accuracy at 73%, which while not ideal, represents a meaningful result given the challenges of the dataset.

As anticipated, the CNN model outperformed the other models, especially the traditional algorithms. However, the traditional learning methods performed better than expected, especially SVM which surpassed the vanilla RNN despite its limitations in capturing complex patterns. This outcome suggests that SVM may have been better suited to the dataset's characteristics, even without the advanced pattern recognition capabilities of neural networks.

A key factor affecting performance across all models was the uneven class distribution of the original dataset, which contributed to difficulties in generalization and led to overfitting. In the future, improvements such as better hardware for faster computation and a larger, more balanced dataset with more patient variety could enhance model performance, reduce overfitting, and lead to higher accuracy.

Model	**Validation Accuracy**	**Training Time**	**Brief Description**
SVM	59.5%	~30 mins	Radial Basis Function (RBF) kernel.
Logistic Regression	52.1%	~3 mins	Max iteration = 1000.
CNN	73.1%	~35 mins	5 Layers using convolution, batch norm, max pooling, and dropout.
RNN	59%	~13 mins	5 Layers with reshape images, LSTM, dense and dropout
Attention RNN	60%	~30 mins	2 LSTM layers, an attention layer, two dense layers, and dropout layers

6. Conclusion

This study evaluated the performance of traditional machine learning algorithms and neural network architectures in detecting brain hemorrhages from CT images. Among the five models tested—Support Vector Machine (SVM), Logistic Regression, Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), and Attention-based RNN—CNN achieved the highest accuracy at 73.1%, demonstrating its ability to capture complex spatial patterns. Traditional models like SVM also performed reasonably well, with the RBF kernel achieving 59% accuracy, surpassing both Logistic Regression and the vanilla RNN. However, despite incorporating attention mechanisms, the Attention-RNN model showed only marginal improvement over the standard RNN, highlighting challenges in differentiating subtle differences in the dataset.

A major limitation encountered across all models was overfitting, particularly in the neural network architectures, as evidenced by significant discrepancies between training and validation accuracies. The uneven class distribution in the dataset further contributed to this issue, emphasizing the need for more balanced and diverse data. While data augmentation and regularization techniques helped mitigate overfitting to some extent, they were insufficient to fully address the gap in generalization performance.

These findings underscore the potential of CNNs for brain hemorrhage detection, while also pointing to areas for improvement in neural network training, particularly for sequential data models like RNNs. Future work could focus on expanding the dataset, enhancing preprocessing strategies, and exploring more advanced architectures such as transformers or hybrid models to improve accuracy and generalization. Furthermore, better hardware and computational resources could enable more comprehensive hyperparameter tuning, leading to potentially better results.

References

[1] Brain bleed, hemorrhage (Intracranial hemorrhage). (2023, December 4). Cleveland Clinic. https://my.clevelandclinic.org/health/diseases/14480-brain-bleed-hemorrhage-intracranial-hemorrhage

[2] Caceres, J. A., & Goldstein, J. N. (2012). Intracranial hemorrhage. *Emergency Medicine Clinics of North America*, *30*(3), 771–794. https://pmc.ncbi.nlm.nih.gov/articles/PMC3443867/#:~:text=Intracerebral%20hemorrhage%2C%20or%20ICH%2C%20is,States%5B1%E2%80%933%5D

[3] Parizel, P. M., & Philips, C. D. (2020). Traumatic Neuroemergency: Imaging Patients with Traumatic Brain Injury—An Introduction. In *IDKD Springer series* (pp. 77–92).

https://www.ncbi.nlm.nih.gov/books/NBK554351/

[4] Liu, Z., Wang, C., & Zhang, J. (2019). Deep learning applications in brain tumor classification using convolutional neural networks. *Medical Image Analysis*, 58, 101567. https://www.sciencedirect.com/science/article/abs/pii/S1361841519301070?via%3Dihub

[5] Oktay, O., Schlemper, J., Le Folgoc, L., et al. (2018). Attention U-Net: Learning Where to Look for the Pancreas. *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*. 506-513. https://arxiv.org/abs/1802.06955.

[6] Peng, X., Zhang, L., & Liu, Y. (2018). Deep learning-based computed tomography image segmentation and volume measurement of intracerebral hemorrhage. *Frontiers in Neuroscience*, 12, 965680. https://www.frontiersin.org/journals/neuroscience/articles/10.3389/fnins.2022.965680/full

[7] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2015. [Online]. Available: https://arxiv.org/abs/1409.0473

[8]J. Chorowski, D. Bahdanau, D. Serdyuk, K. Cho, and Y. Bengio, "Attention-based models for speech recognition," in *Advances in Neural Information Processing Systems,* vol. 28, pp. 577-585, 2015.

[9]I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Advances in Neural Information Processing Systems,* vol. 27, pp. 3104-3112, 2014. [Online]. Available: https://arxiv.org/abs/1409.3215

[10]Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature,* vol. 521, no. 7553, pp. 436-444, 2015. [Online]. Available: https://doi.org/10.1038/nature14539

[11]K. Xu, J. Ba, R. Kiros, et al., "Show, attend and tell: Neural image caption generation with visual attention," in *Proc. International Conference on Machine Learning,* 2015. [Online]. Available: https://arxiv.org/abs/1502.03044

[12] Our Python Code