ISTANBUL TECHNICAL UNIVERSITY
FACULTY OF COMPUTER AND INFORMATICS

MEDICAL IMAGE SEGMENTATION IN MRI SCAN IMAGES

Graduation Project

Cem Yusuf Aydoğdu
150120251

Program: Computer Engineering
Department: Computer and Informatics

Advisor: Assoc. Prof. Dr. Gözde ÜNAL

December 2016

ISTANBUL TECHNICAL UNIVERSITY
FACULTY OF COMPUTER AND INFORMATICS

MEDICAL IMAGE SEGMENTATION IN MRI SCAN IMAGES

Graduation Project

Cem Yusuf Aydoğdu
150120251

Program: Computer Engineering
Department: Computer and Informatics

Advisor: Assoc. Prof. Dr. Gözde ÜNAL

December 2016

Özgünlük Bildirisi

1. Bu çalışmada, başka kaynaklardan yapılan tüm alıntıların, ilgili kaynaklar referans gösterilerek açıkça belirtildiğini,
2. Alıntılar dışındaki bölümlerin, özellikle projenin ana konusunu oluşturan teorik çalışmaların ve yazılım/donanımın benim tarafımdan yapıldığını

bildiririm.

İstanbul, 30.12.2016

Cem Yusuf Aydoğdu

MEDICAL IMAGE SEGMENTATION IN MRI SCAN IMAGES

(SUMMARY)

In the guiding light of science and technology, humankind proceeds to overcome immanent and subsequently emerging problems about its health. Area of medical technologies is one major research and application area for this purpose, as well as medical imaging is one of the many important steps taken through this direction. As a quick summary of medical imaging techniques, crucial advances occurred during the previous century, for instance utilization of nuclear magnetic resonance, ultrasound and computer in medical imaging technologies. Consequently, computational analysis and applications in various medical imaging formats emerged under the definition of medical image processing.

Topic of the project is about identification and discrimination of ischemic stroke lesions among healthy tissues, in three dimensional magnetic resonance volumes. Medical aspects about the ischemic stroke disease are investigated briefly in the "Introduction" section.

Initially, literature about solution domain which is proposed by field of computer science was examined. By the result of this exploratory study, methods and tools for the project were designated. These results are given in the "Introduction" part. Academic knowledge that was gained through all stages of the project is provided in the "Background Information and Methods" section. In the "Analysis and Modeling" section, fundamental logic and structure of the data is analyzed and the project architecture is modeled.

First, problem is addressed as a computer vision segmentation problem. For this reason, "GrowCut" algorithm which provides semi-automatic segmentation in multidimensional image data was implemented and tested. Then, in order to obtain fully automatic segmentation, problem is addressed as a machine learning classification problem. Therefore, k-nearest neighbors", "linear classification", "support vector machines", "random forests" and "convolutional neural networks" algorithms were utilized for the project. Technical details about these methods are given in the "Design, Implementation and Test" section.

Outcomes of "GrowCut" algorithm were compared with ground truth images in a slice-wise way. Manual parameter adjustments were performed to improve results from GrowCut algorithm. Machine learning algorithms were evaluated with "k-fold cross validation" approach and various performance metrics such as "sensitivity", "specificity" and "accuracy". Detailed information about metric results are provided in the "Experimental Results" part of this report.

In the learning based models, "convolutional neural network" method was observed with nearly 89% accuracy. "Random forest" algorithm was performed with 81% accuracy score, whereas "k-nearest neighbor" was obtained 80% accuracy. In conclusion, "convolutional neural network" was observed as the best performing algorithm among other models.

MEDICAL IMAGE SEGMENTATION IN MRI SCAN IMAGES

(ÖZET)

İnsanlık, sağlık konusunda geçmişten beri var olan ve sonradan ortaya çıkan problemleri bilim ve teknolojinin ışığında aşmaya devam etmektedir. Sürekli olarak gelişen, bilgisayar ve bilişim sistemlerinin yoğun olarak kullanıldığı tıpla ilgili teknolojiler bu ilerleyişe yardımcı olmaktadır. Tıbbi verilerin incelenmesi, işlenmesi ve anlamlandırılması bahsedilen teknolojilerden bir örnektir.

Projede yapılan çalışma üç boyutlu manyetik rezonans verilerinde iskemik inme hastalığına dair doku bozukluklarının belirlenmesi ve hastalıklı dokuların sağlıklı dokulardan ayırt edilmesi üzerinedir. Bu hastalık beyni besleyen kan damarlarında tıkanma veya zedelenme sonucu beyin dokularının geçici veya kalıcı olarak hasarlanması olarak özetlenebilir [1]. İstatistiklere göre bu hastalık her yıl dünya çapında yaklaşık 15 milyon kişiyi etkilemekte, 6 milyon kişinin ölümüne sebebiyet vermekte ve 5 milyon kişide kalıcı hasar bırakmaktadır [2]. Ayrıca bu rahatsızlık dünya genelinde ileri yaştaki insanlar için ikinci en sık rastlanan ölüm sebebiyken, genç yaştaki insanlar için beşinci en sık rastlanan ölüm sebebidir [2].

Projede yapılan ön çalışma, problemin çözümüne dair bilgisayar bilimleri kapsamında daha önce yapılan çalışmaların araştırılması olarak özetlenebilir. Ön çalışmanın yardımıyla projede kullanılacak yöntemler ve araçlar belirlendi. Bu bilgilere raporun "Introduction" bölümünden ulaşılabilir. Projede ihtiyaç duyulan kuramsal bilgiler raporun "Background Information and Methods" kısmında yer almaktadır. Raporun "Analysis and Modelling" kısmında, projede kullanılan manyetik rezonans görüntüleme tekniğiyle elde edilmiş veriler analiz edildi, geliştirilecek sistem modellendi.

Problem, hem bilgisayarla görü, hem de makine öğrenmesi alanlarının perspektifinden incelendi. Problem bilgisayarla görü alanı çerçevesinde 'segmentasyon', makine öğrenmesi altında 'sınıflandırma' olarak modellendi. Eldeki verilerin niteliği ve yapısı gereği makine öğrenmesi algoritmaları üzerine yoğunlaşıldı.

Uygulama aşamasında ilk olarak, çok boyutlu görsel verilerde yarı-otomatik segmantasyon imkanı sağlayan "GrowCut" algoritması gerçeklendi ve projenin verileriyle test edildi. Daha sonra, tam otomatik ayırt etme amacıyla makine öğrenmesi alanında sınıflandırma amaçlı kullanılan çeşitli modeller üzerinde çalışıldı. Veriler öğrenme tabanlı yöntemler için düzenlendi. Sırasıyla "k-nearest neighbors", "linear classification", "support vector machines", "random forests" ve "convolutional neural networks" yöntemleri projeye uyarlandı. Uygulanan algoritmalarla ilgili teknik detaylar "Design, Implementation and Test" bölümünde açıklanmıştır.

Bilgisayarla görü alanından "GrowCut" algoritması ile elde edilen segmentasyon sonuçları, gerçek referans(ground truth) verileriyle görsel olarak karşılaştırıldı. Algoritmanın sonuçları parametre ayarlarıyla iyileştirilmeye çalışıldı. Projede kullanılan makine öğrenmesi algoritmaları ise "k-fold cross validation" tekniği yardımıyla değerlendirildi, algoritmaların performansları hassasiyet (sensitivity), belirlilik (specificity) ve kesinlik

(accuracy) metrikleri ile ölçüldü. Bu metrikleri kullanarak elde edilen sonuçlar detaylı olarak raporun "Experimental Results" kısmında açıklanmıştır.

Makine öğrenmesi modelleri arasında "convolutional neural networks" yaklaşık %89 kesinlik oranı elde etti. Bu algoritmayı "random forest" %81 kesinlikle ikinci olarak, "k-nearest neighbors" ise %80 kesinlikle üçüncü olarak takip etti. Sonuç olarak, bahsi geçen performans metriklerinin yardımıyla, kullanılan yöntemlerin arasında "convolutional neural networks" en başarılı yöntem olarak gözlemlendi.

TABLE OF CONTENTS

1. Introduction

In the guiding light of science and technology, humankind proceeds to overcome immanent and subsequently emerging problems about its health. Area of medical technologies is one major research and application area for this purpose, as well as medical imaging is one of the many important steps taken through this direction. As a quick summary of medical imaging techniques, crucial advances occurred during the previous century, for instance utilization of nuclear magnetic resonance, ultrasound and computer in medical imaging technologies. Consequently, computational analysis and applications in various medical imaging formats emerged under the definition of medical image processing.

Problem subject to this project is identification of stroke lesions in brain, specifically ischemic stroke. Stroke can be defined as perpetual or temporary injury in brain tissues due to stoppage or breakage of vessels that transport oxygen and various nutritions to brain [1]. Ischemic stroke is a type of stroke which is caused by obstruction of a blood vessel in the brain, in most cases a blood clot [1]. According to statistics, stroke impacts 15 million individual, causes approximately 6 million casualties and 5 million disabilities in global [2]. Moreover, it is the second most common mortality factor among elders and the fifth most common mortality factor among youngers [2]. Also, ischemic stroke consists %80 of entire stroke incidents [2]. Project topic and problem were derived from 2015 Ischemic Stroke Lesion Segmentation (ISLES) challenge which was created for 18th International Conference on Medical Image Computing and Computer Assisted Intervention (MICCAI 2015) [4].

According to Withey and Koles, previous work about segmentation of disease symptoms or lesions in medical images can be examined in three phases, where each phase corresponds to an augmentation of algorithmic complexity [3]. First phase consists of simplex heuristic applications such as thresholding, region growing and edge tracking [3]. Along with the demand of automatic methods, second phase studies recommend optimization and uncertainty models instead of heuristics, for instance deformable models, c-means clustering and neural networks [3]. Yet, third phase methods which depend on further high level information are emerged, such as atlas based segmentation, rule based segmentation and other shape based models [3].

Another study by Rekik et. al. indicate that most of old work done for ischemic stroke lesion segmentation utilized simple thresholding techniques with a mixture of statistical methods [5]. They also report that insufficient analysis about biological and imaging based problems is a significant problem. They advise researchers to incorporate different points for instance variation of ischemic stroke lesions patterns in different patients, progression mechanisms of the ischemic stroke disease, detailed clinical information about patients [5].

Since the project is based on the ischemic stroke lesion segmentation challenge described above, articles that are accepted from this challenge are also examined. It is observed that most of the studies utilized several variations of random forest classification approach, whereas the two most successful studies used convolutional neural networks and fuzzy c-means approaches.

In this project, problem is addressed from two different but similar perspectives in the area of computer science and engineering. Initially the problem is examined as a computer vision segmentation problem, then it is inspected as a machine learning classification problem. First, GrowCut algorithm from computer vision area is implemented and tested. Then, various machine learning models are utilized for the problem. Methods are explained and results are presented. Finally, outcomes from different methods are compared.

Administrative details about the project such as definition and plan are explained in the Section 2. Academic knowledge that are gained through all stages of the project are provided in Section 3. Examination of the problem and proposed model for the project are given in the Section 4. Details about design, implementation and test phases of the project are described in the Section 5. Various experiments about implemented functionalities and their results are explained in Section 6. Review about the project and suggestions for further improvement are given in the Section 7.

2. Project Definition and Plan

2.1 Project Description

Project topic can be clarified as application and evaluation of various image segmentation methods using computer vision and machine learning algorithms on 3D structural brain volume data which are obtained through the instrument of magnetic resonance imaging (MRI) technique, in order to detect and mark ischemic stroke lesions.

Essential success criterion of the project is to determine stroke lesions correctly in the brain tissue. Definition of correctness is determined by different metrics, for instance precision and recall. These metrics are explained in the next section.

2.2 Project Plan

2.2.1 Project Scope

Scope of the project is limited to segmentation of ischemic stroke lesions, in the medium of 3D structural volume data which is obtained by magnetic resonance imaging.

Also, scientific examination of the ischemic stroke disease or inspection of project outcomes from a physician's perspective is beyond the scope of this project.

2.2.2 Estimates About The Project

Total estimated time for the project is approximately 40 weeks,

Estimated work load of the project can be expressed in three main topics:

- Preparation and preprocessing of the data
- Realization of different segmentation methods on the data
- Evaluation and comparison of results

2.2.3 Project Resources

Project sources include articles for theoretical basis of the project, specific software libraries with their documentations for selected programming environment and dataset of the project. Scientific articles for the project were obtained from various online and offline sources. As developing environment, Python language was used under Ubuntu Linux 14.04 platform. Software library sources of the project are given below in the Table 2.1 with their purposes.

Table 2.1: List of python packages used in the project

Package	**Purpose**
Matplotlib	2D visualization
Scikit-learn	Machine learning
Nibabel	To use NifTI data format
IPython	Development in Python language
Numpy	Scientific computing

Matplotlib is an open source software library that provides 2D visualization tools for bar plots, images, histograms etc. [6]. Scikit-learn is a package that provides a simple interface for commonly used supervised and unsupervised machine learning algorithms in Python environment [7]. Another Python module Nibabel supplies instruments to read or write various medical imaging formats such as ANALYZE, NIfTI1, NIfTI2 and DICOM [8]. IPython package adds interactive shell and web-based development environment features to Python [9]. NumPy module provides essential tools for scientific computation, such as multidimensional arrays and linear algebra tools [10].

As data source for the project, sub-acute ischemic stroke lesion segmentation dataset of the ISLES challenge was used. This source will be examined in detail in the chapter 4.1.

2.2.4 Risks & Risk Management

There are several risks are involved with the project, for instance delay of the work packages, change in requirements and dense personal schedule may alter time plan. Also, possible inability or inaccuracy of planned methods or algorithms may affect the project.
To minimize these risks, general situation about the project was evaluated regularly. Risk mitigation table is shown below in the Table 2.2.

Table 2.2: Risk mitigation table

Risk	Importance Factor (Over 5)	Probability
Change in the requirements	4	%25
Delay of work packages	4	%35
Inaccuracy in proposed methods	4	%30

2.2.5 Time Plan

Milestones	Start Date	(weeks)	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40
Determination of requirements	01.03.2016	2																																								
Analyse of the problem	14.03.2016	2																																								
Research in the problem domain	21.03.2016	7																																								
Research about image segmentation	18.04.2016	7																																								
Installation & practice for required tools	16.05.2016	3																																								
Practice about image segmentation	30.05.2016	4																																								
Development of the project	20.06.2016	18																																								
On-going research during project	20.06.2016	18																																								
On-going test of the project	08.08.2016	12																																								
Final documentation about project	10.10.2016	8																																								

Table 2.3: Gantt chart of the project plan

3. Background Information and Methods

3.1 Magnetic Resonance Imaging

Magnetic resonance imaging (MRI) is an imaging technique which is used for diagnostic, treatment and monitoring purposes in medical cases. Magnetic resonance imaging is a non-invasive technique; also it does not use any ionizing radiation source such as x-rays [11].

Magnetic resonance imaging exploits distinct magnetic properties of different tissues, by initially aligning magnetic moments of hydrogen atoms inside tissues with a strong constant magnetic field, then disturbing these atoms by electromagnetic waves at their resonance frequency, measuring their response and interpreting these results as images via transformation from frequency domain to spatial domain [12]. Quantification of various subatomic and interatomic properties results in different modalities of magnetic resonance imaging, such as FLAIR, T_1, T_2 and MRI [12]. These different modalities are used for disambiguating distinct types of tissues in the brain such as fat, cerebrospinal fluid, white matter and cortex [13].

3.2 Neuroimaging Informatics Technology Initiative (NIfTI) Data Format

Neuroimaging Informatics Technology Initiative is a working group which is supported by various institutes that aims augmentation of instruments used by neuroimaging research communities and resolution of prevalent complications among existing tools such as compatibility [14]. NIfTI-1 is a format standard for storage of different types of data.

All MRI data concerned about this project is in the NIfTI-1 data format. Nonetheless, low level details about NIfTI-1 data format are handled by a third party software library in the development phase of the project, therefore those details are skipped in this report.

3.3 Computer Vision

Field of computer vision can be explained as computational representations of abilities that exist in human visual system, and utilization of human specific or above human visual abilities in machines [15]. Example research areas of computer vision include image enhancement, feature extraction, feature detection, object segmentation and object recognition. In the project, an image segmentation algorithm is applied in order to differentiate lesion and non-lesion regions.

3.3.1 GrowCut

GrowCut is an algorithm which uses cellular automaton model to perform semi-automatic image segmentation [16]. The algorithm is able to work with multidimensional images and multiple labels [16]. These abilities provide a wide usage area for the algorithm. Furthermore, users can interfere the automatic labeling process directly, by appointing appropriate labels for wrongly labeled pixels [16]. The algorithm is adaptable, easy, and able to overcome difficult problems [16]. Even though the algorithm has a significant drawback from manual initialization step, there are several approaches to automatize this step.

Algorithm initiates with a state set for each "cell", where each cell corresponds to a pixel or voxel in the image. These sets contain label, strength and feature values for each cell. Main logic behind the algorithm is as following: first, user manually sets labels of some cells in the image, and then the algorithm automatically determines labels for other pixels by using a cellular automaton. In the automatic phase, labels at initialized pixels are expanded in non-labeled areas like bacteria in a nutritious medium. When these labels confront, they are stabilized with the help of some parameters, such as their pixel values and strength values. Pseudocode of this algorithm is given below in the Figure 3.1.

```
procedure GROWCUT
    Initialize feature with intensity values
    Initialize label with foreground and background seeds
    Initialize str with strengths
    Initialize next seeds label_next
    Initialize next strengths str_next
    for each iteration do
        for each cell do
            Set label_next[cell] ← label[cell]
            Set str_next[cell] ← str[cell]
            Obtain neighbors of cell
            for each neighbor in neighbors do
                Calculate dist ← abs(feature[neighbor] − feature[cell])
                Calculate value ← g(dist) ∗ str[neighbor]
                if value > str[cell] then
                    label_next[cell] ← label[neighbor]
                    str_next[cell] ← value
                end if
            end for
        end for
        Set label ← label_next
        Set str ← str_next
    end for
end procedure
```

Figure 3.1: Pseudocode for the GrowCut algorithm.

3.4 Machine Learning

Area of machine learning is concerned about foreseeing characteristics about unseen data with the help of prior knowledge which is extracted from existing data. Machine learning topics can be examined in two branches: supervised learning where the aim is to infer a matching rule between input data and outcome preset labels, and unsupervised learning which corresponds to derivation of patterns in the unlabeled data [17]. Both of these branches has multiple subsets of research and application areas such as classification, regression in supervised learning and clustering in unsupervised learning [17]. In classification problems, output of the objective is discrete integer number values, for instance distinguishing lesion and non-lesion labels of voxels in given MRI sequences, as in this project. In regression problems, output is a continuous number. As an example, estimating price of a house by using several features of the house is a regression problem.

Learning procedure in the supervised learning approach can be mathematically modeled as $y = f(x)$, where y is desired output such as discrete values to indicate labels or continuous values to refer a prediction. Goal is to learn function f from data, such that function f maps given input x to corresponding y. This goal can be achieved by different methods, for instance by defining a "cost function" which measures difference between expected results and obtained results, and optimizing the function to minimize this deviation.

Since the problem of the project is a classification problem from the perspective of machine learning, only certain classification algorithms in the supervised learning branch of the machine learning area are studied.

3.4.1 K-Nearest Neighbor Classification

K-nearest neighbor algorithm is an early instance in the machine learning area. Main principle of the algorithm is to classify the data according to labels of *k* adjoining or close data, presuming that data instances are identically and independently distributed [18]. Degree of closeness is measured by a distance metric such as euclidian distance. K-nearest neighbor classification algorithm is a non-parametric and instance-based learning algorithm [17]. Visual explanation is given below in the Figure 3.2.

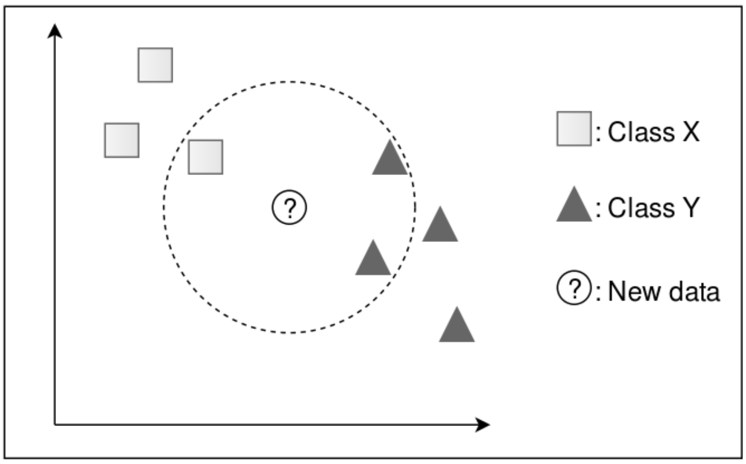


Figure 3.2: Illustration of *k*-nearest neighbors classification with $k = 3$. New data is classified with label Y, since majority of its three neighbors belong to class Y.

3.4.2 Linear Classification

In linear classification, a linear function which is composed of weighted sum of features is used to discriminate samples with different labels, under the presumption that classes can be easily separated a linear function [19]. It is used commonly due to its mathematically uncomplicated nature in terms of both implementation and understanding [19]. An instance is given below in the Figure 3.3.

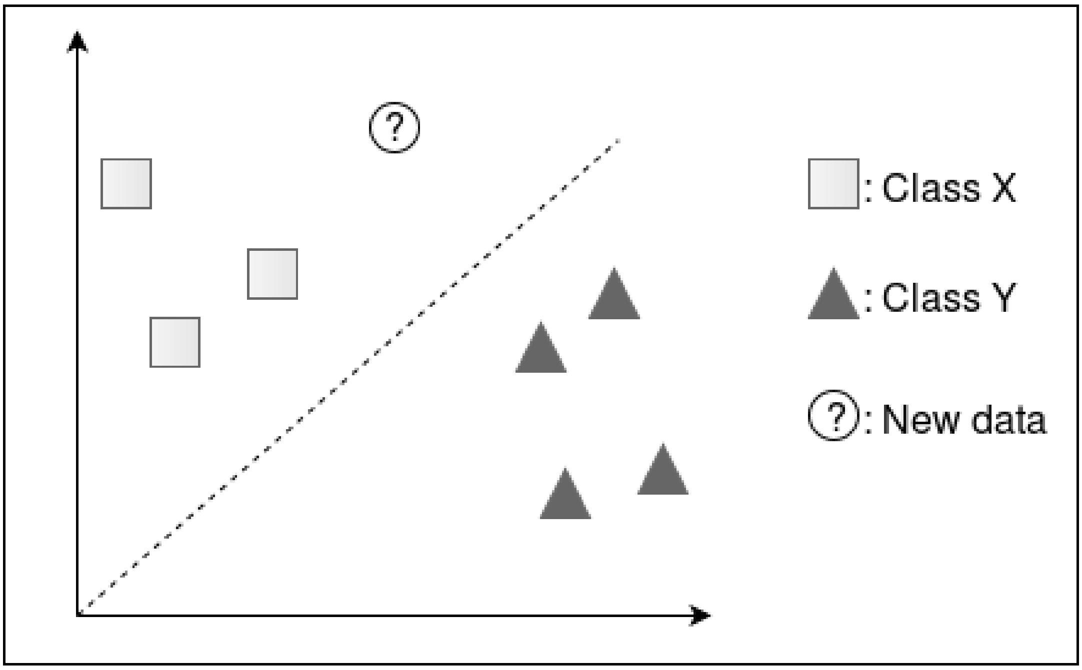


Figure 3.3: A simple example linear for linear classification. Dotted line shows the separating linear model. In this case, new data is classified as X.

3.4.3 Support Vector Machines

Support vector machines are another set models in the supervised learning branch of machine learning. Working principle of support vector machines is also based on dividing "separable" classes by using linear or polynomial mathematical models, or kernels in other words [20]. But, different than a simple linear classifier, support vector machines also try to optimize the separation by maximizing "margins" between separated classes [20]. This principle can be visualized in Figure 3.4 below.

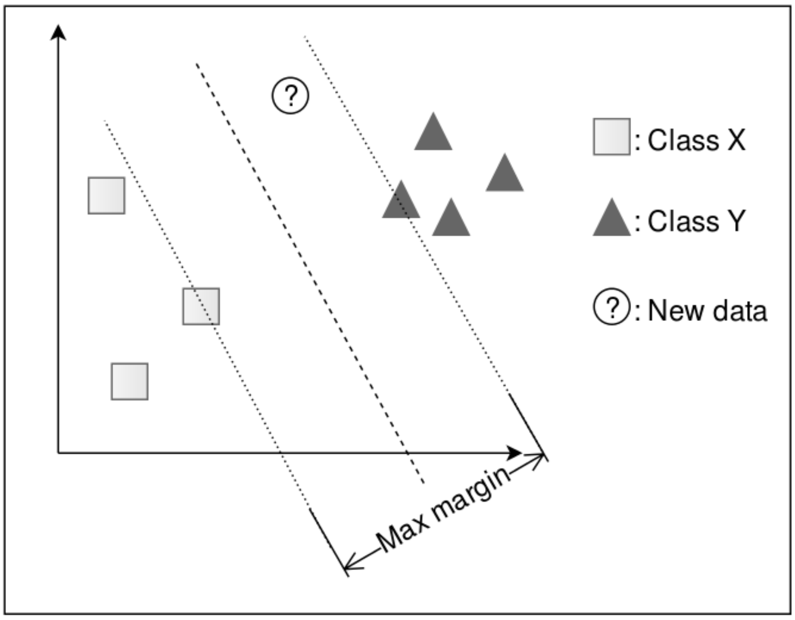


Figure 3.4: Support vector machine finds the best separator by maximizing the margin.

3.4.4 Random Forests

Random forest is one of the “ensemble” learning methods, which indicates utilizing combinations of various “decision tree” models in order to achieve better performance [17]. Ensemble learning is the technique of mixing distinct results from different learning models [17]. Decision tree is also a classification or regression method that can be modeled as a tree to divide input space recurrently in order to perform classification or regression [17]. An example random forest is given below in the Figure 3.5.

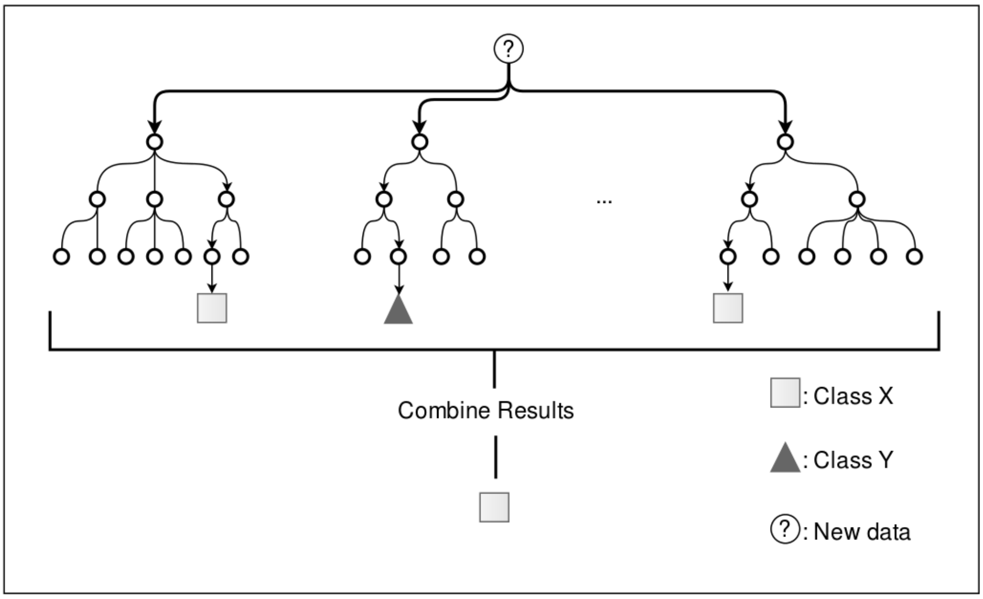


Figure 3.5: Random forest combines results of various decision trees.

3.4.5 Artificial Neural Networks

Artificial neural networks are another computational approach which are inspired by operation principle and architecture of neurons in the brain. An artificial neural network contains atomic calculation modules as “neurons” in layers and connections between these calculation modules to simulate “synapses”. Every neuron includes multiple input values, a bias value, an output value, and an activation function to calculate output by using its weighted inputs. Purpose of the bias value is to alter output value independent from weighted inputs. An illustration of the neuron structure is given below in the Figure 3.6. Typical architecture of a neural network is given below in Figure 3.7.

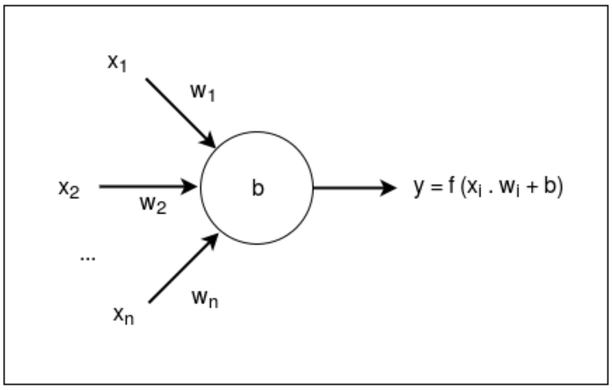


Figure 3.6: Neuron calculates an output value depending on its parameters.

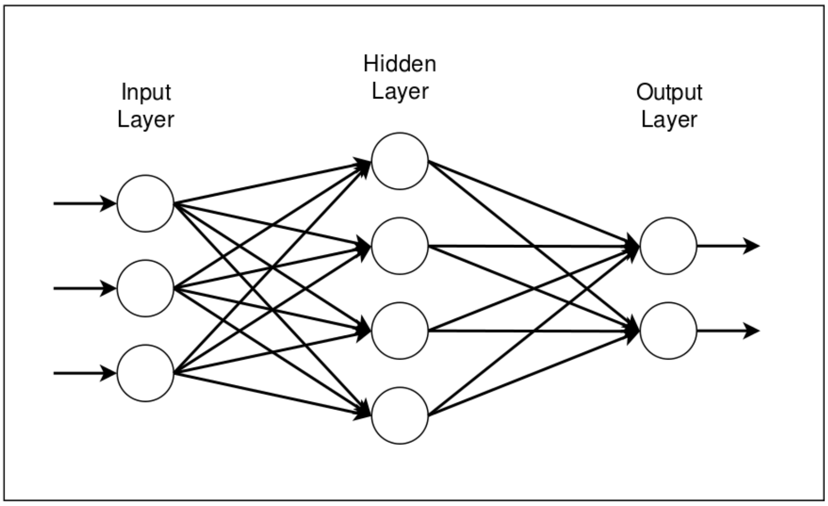


Figure 3.7: A simple neural network. Neurons are settled in layers and connections between neurons are directed to forward.

In order to utilize a neural network for a classification problem, all neurons in the network should “learn” their weight and bias parameters. Aim is to obtain correct label at the output, for a given input. Similar to other machine learning models, a cost function is determined for the network model, which will be used to optimize network parameters.

Neural network learns its weight and bias parameters by optimizing a predetermined cost function which measures difference between desired and obtained results. This optimization procedure is performed in the training phase by an algorithm named as gradient descent. Gradient descent aims to minimize cost function in an iterative way by calculating gradients in the cost function with respect to weight and bias parameters. Further information about optimization step of the algorithm by gradient descent can be obtained from Deep Learning book by Goodfellow et. al. [21].

3.4.5.1 Convolutional Neural Networks

Architectures of the regular networks are obviously not convenient for high dimensional data such as images. This is because layer structures of typical neural networks are one-dimensional. Application of typical neural networks to images requires high numbers of parameters, which will result as increased complexity. Therefore, convolutional neural networks are commonly used in application about images. In short, convolutional neural networks are a special form of neural networks which utilize convolution operation in some layers, rather than basic dot-products [22].

Convolutional neural networks utilize three important tools: local receptor fields, shared weight and biases, pooling [23]. Local receptor fields represent small fields with shared weight and biases, that perform convolution operations to regions of image by a sliding window approach [23]. Weights and biases are shared inside distinct local receptor fields, that allows a local receptor field to apply the same convolution operation to the whole image [23]. These shared weights of local receptor fields are called “kernels”, outputs of local receptor fields are named as “feature maps” [23]. An example local receptor field is given below in the Figure 3.8.

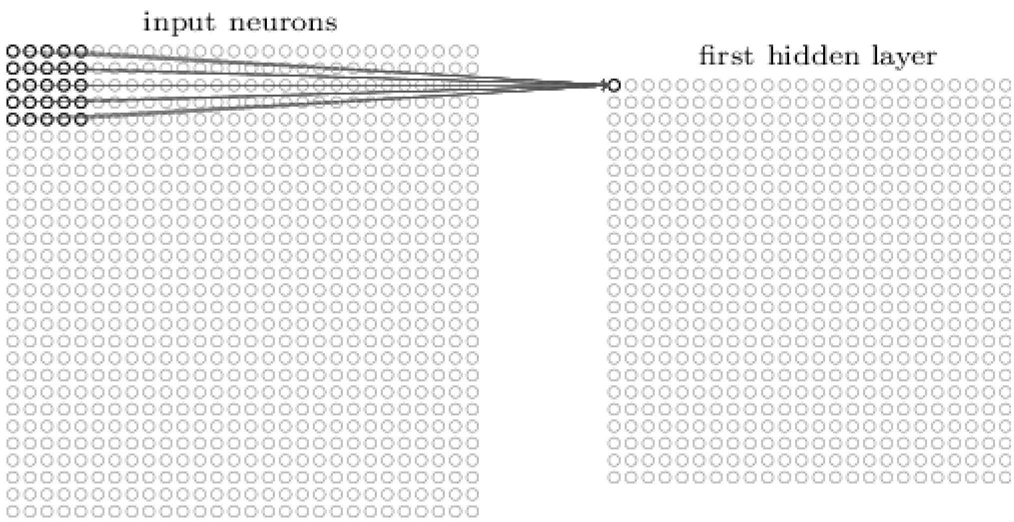


Figure 3.8: Operation of a local receptor field. In the left, “kernel” denotes to weights of visible connections from input neurons to first hidden layer. Also, first hidden layer in the right corresponds to a “feature map” [23].

Pooling operation is used to downsample feature maps in order to transfer only significant information to next layers and to reduce number of parameters in the network, commonly by taking the maximum values in fields [22]. An example of pooling is given below in the Figure 3.9.

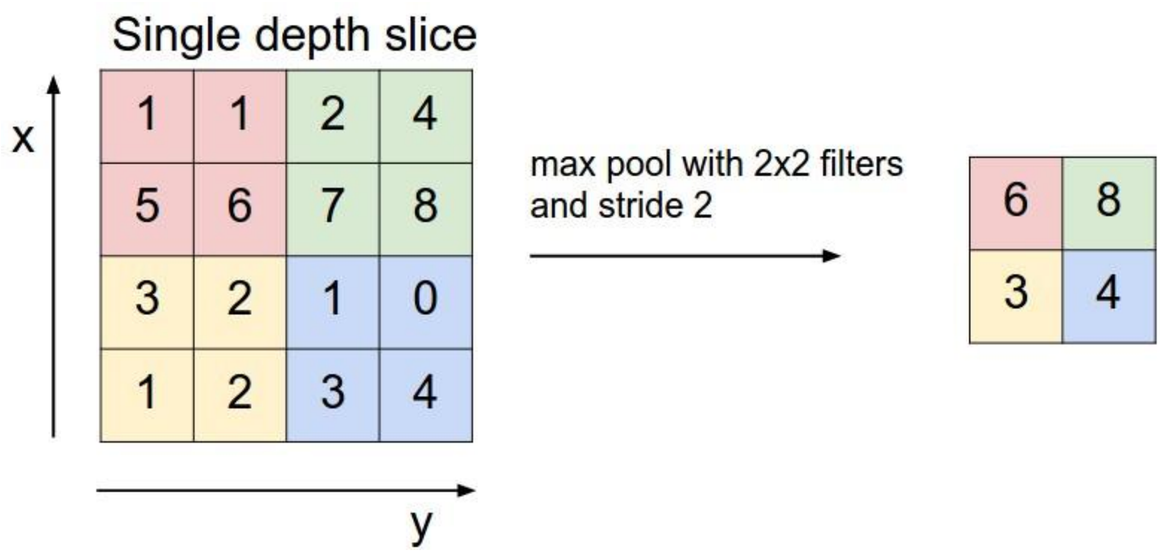


Figure 3.9: Application of max pooling on a feature map [24].

3.5 Performance Evaluation Methods

3.5.1 K-Fold Cross Validation

In K-Fold cross validation, data is split to *k* equal (or nearly equal if sample size is not divisible by *k*) partitions. After that, evaluation is applied *k* times for each fold, where selecting a fold as test data and other folds as training data [25]. An illustration of this approach is given below in the Figure 3.10.

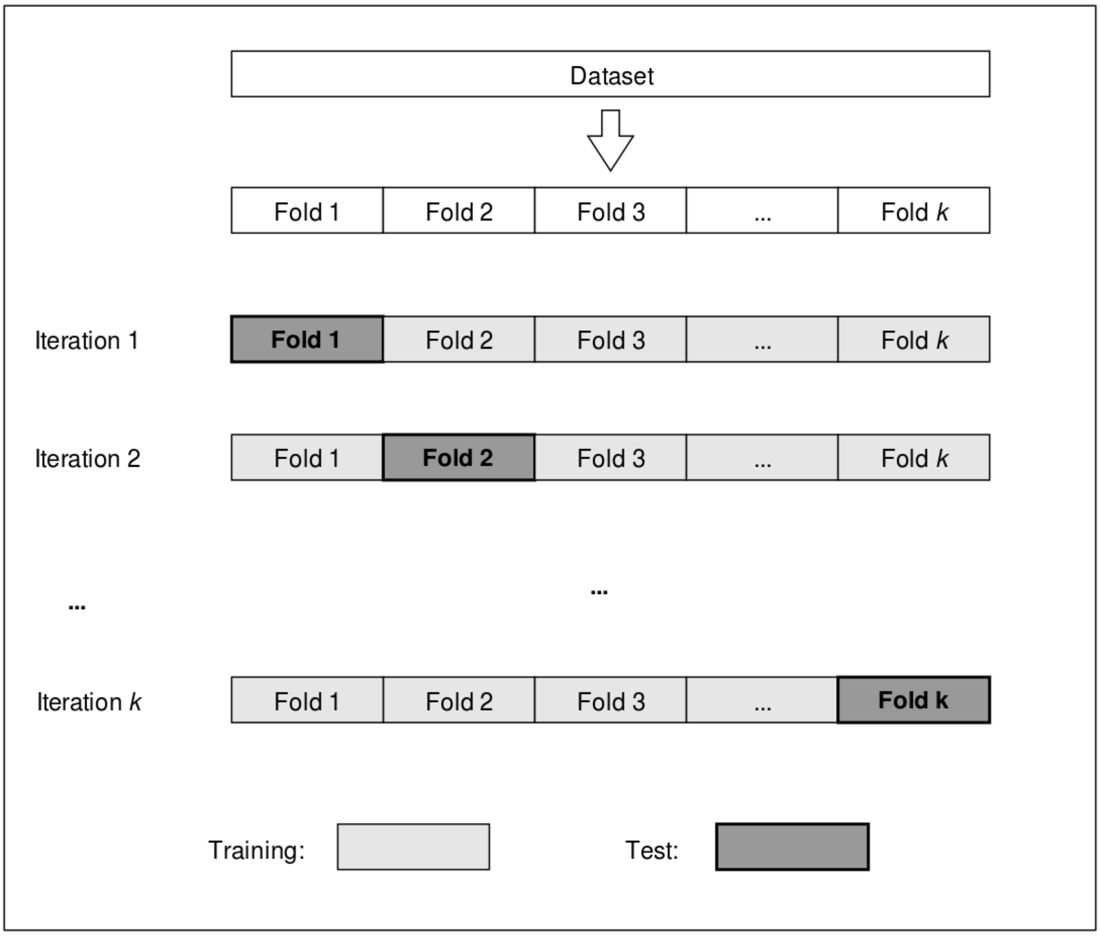


Figure 3.10: Illustration of k-fold cross validation.

3.5.2 Confusion Matrix

Confusion matrix is a table which includes various statistical metrics about classification results, such as true positives, true negatives, false positives and false negative [26]. When the model predicts the label of a sample correctly, it is considered as true positive (TP) if the label is actually positive (lesion) and true negative (TN) if the label is actually negative (non-lesion) [26]. False positive (FP) occurs when the label is actually negative but identified as positive and false negative (FN) denotes when the label is actually positive but predicted as negative [26]. Table 3.1 shows the layout of the confusion matrix used in this project.

Table 3.1: Layout of the confusion matrix used in evaluation

		Actual Label	
		Positive	Negative
Predicted Label	Positive	TP	FP
	Negative	FN	TN

3.5.3 Sensitivity, Specificity and Accuracy

After obtaining confusion matrices, different metrics such as sensitivity, specificity and accuracy were calculated with values from confusion matrices. Sensitivity shows the ability of the model to recognize true positive labels (lesion), specificity denotes the skill of detecting true negatives (non-lesion) and accuracy refers to correctness of the model in both correct positive and negative labels [27].

Sensitivity, specificity and accuracy are statistical metrics used in the evaluation of machine learning algorithms. These metrics are calculated with the help of confusion matrix explained above. Sensitivity shows the ability of the model to recognize true positive labels (lesion), specificity denotes the skill of detecting true negatives (non-lesion) and accuracy refers to correctness of the model in both correct positive and negative labels [27]. Mathematical formulations of these metrics are provided in the Figure 3.11 below.

$$Sensitivity = \frac{TP}{TP+FN} \qquad Specificity = \frac{TN}{TN+FP} \qquad Accuracy = \frac{TN+TP}{TN+TP+FN+FP}$$

Figure 3.11: Formulations of performance metrics [27]

4. Analysis and Modeling

4.1 Preliminary Examination of the Dataset

Dataset of the project was examined after determination of project topic, resources, requirements and study of initial literature search.

Original data source of the 2015 MICCAI Ischemic Stroke Segmentation Challenge consists of two data sources for two different sub-challenges, namely sub-acute ischemic stroke lesion segmentation (SISS) and acute stroke outcome/penumbra estimation (SPES) [4]. Each of these two data sources have two datasets inside, one for training and one for testing. Both data sources are preprocessed by challenge organizers in terms of registration, isotropic spacing and removal of skull [4].

Dataset of the project is determined as the training dataset of the sub-acute ischemic stroke lesion segmentation sub-challenge (SISS). The reason is that only training datasets has ground truth information about lesion tissues.

The dataset of the project contains 3D MRI volumes for 26 patients. For each patient, there exists four different modalities of brain scan volumes including 'T1', 'T2', 'Flair' and 'MRI', and also one ground truth volume data denoting lesion and non-lesion voxels, namely 'OT'. Ground truth volumes are predicated on manual segmentations of experienced medical specialists [4].

An example slice with all modalities from 15^{th} patient in the dataset is given below in Figure 4.1. In this figure, differences between T1, T2, DWI and Flair MRI sequences can be visually seen and the abnormal region with lesion can be perceptible even by non-experts because of significant contrast of intensity. However, this assumption cannot be extended to the entire dataset, for instance Figure 4.2 shows modalities of 18^{th} patient who has small and sparse lesion tissues in various locations.

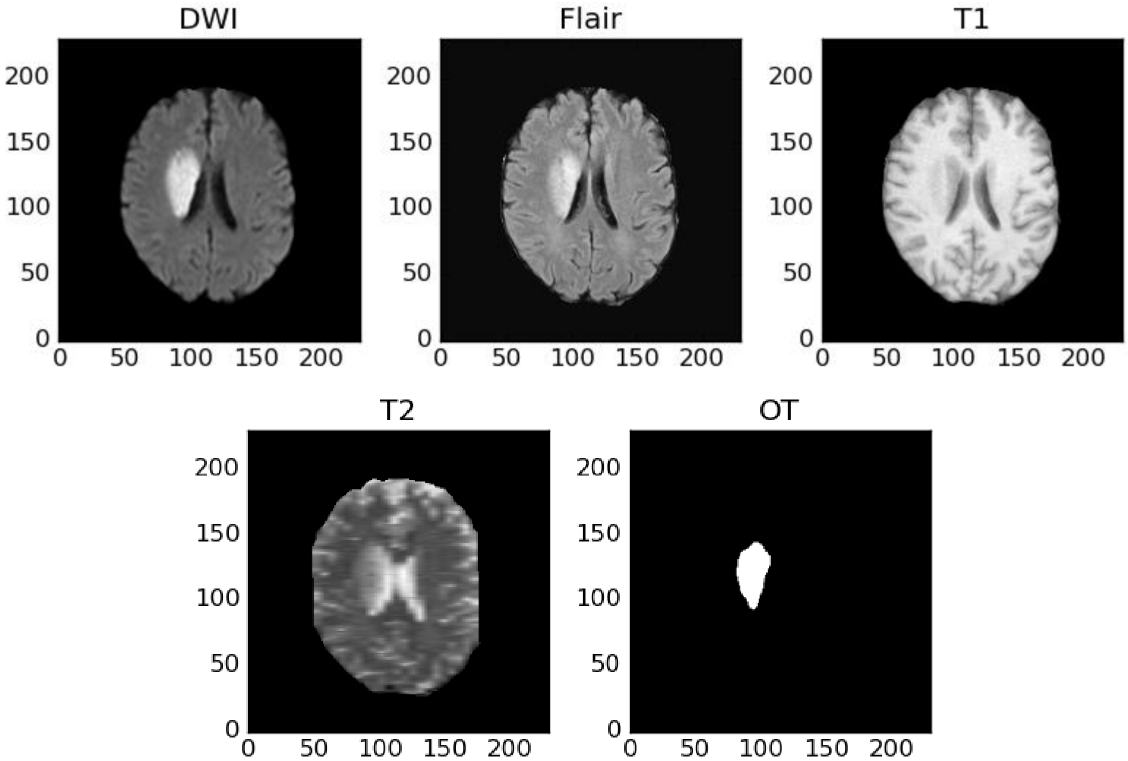


Figure 4.1: 95th axial slice of four MRI modalities and ground truth from 15th patient in the dataset.

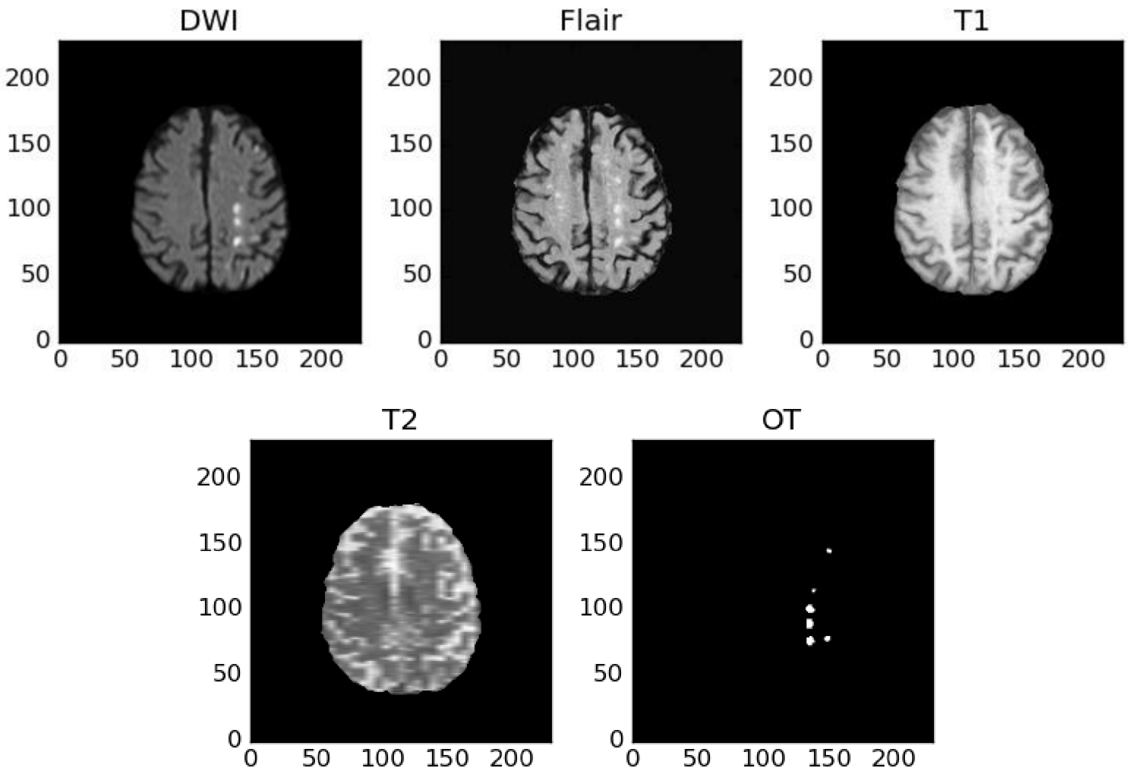


Figure 4.2: As a harder example, 118th axial surface from all modalities of 18th patient.

4.2 Modeling

Object oriented programming approach is not applied in this project. Reasons are as following: object oriented methodology would require additional effort in the modeling phase, and the implementation phase of the project is primarily concerned of prototyping required work packages quickly.

Rather than object oriented approach, project model is composed as various modules, for instance data manipulation module, visualization module. Each module consist several functions to perform required work. These modules are explained in the next section.

5. Design, Implementation and Test

5.1 Design

Design of the project is based on modules in Python. A module can be defined as a container which consists definitions and statements [28]. Main module of the project contains several sub-modules, each with different set of sub-modules or functionalities. In the implementation phase, components of modules are also used together with third party libraries. Architecture of the design is given below in the Figure 5.1. In this figure, filled and empty bullet points represent modules and sub-modules, whereas entities without bullet points denote functions. Documentations about functions are provided within the code.

```
• data
      ○ io
              getFilePath
              loadNiftiData
              saveNiftiData
      ○ process
              getPatch
              zeroPatchControl
              normalizeData
              getDataStats
              calculateBrainmask
              flatPatches
              applyBrainmask
• visualization
        showSingleSlice
        showMultiSlices
        showAnimateVolume
        showHistogram
• growcut
        neighborhood3d_vonNeumann
        growcut_3d
• ml
      ○ eval_sklearn
              eval_sklearnModel
              eval_sklearnModelKFoldCV
      ○ eval_tflearn
              labelToNormal
              labelToOneHot
              construct3DCnn
              train_tflearnModelKFoldCV
              eval_tflearnModelKFoldCV
```

Figure 5.1: Module architecture of the project design.

5.2 Implementation

5.2.1 Loading and Visualizing Data

Loading and visualizing the 3D MRI data format was the first step in implementation process. In order to handle loading and saving operations on NIfTI data format, "Nibabel" package is used. For visualizing 2D slices in the 3D data, "pyplot" sub-module in the "matplotlib" library is adopted.

Visualization functions include viewing single slices separately and displaying multiple slices as animation or side by side. In addition, histogram of a desired slice can be visualized. Examples are given below in Figures 5.2, 5.3, 5.4 and 5.5.

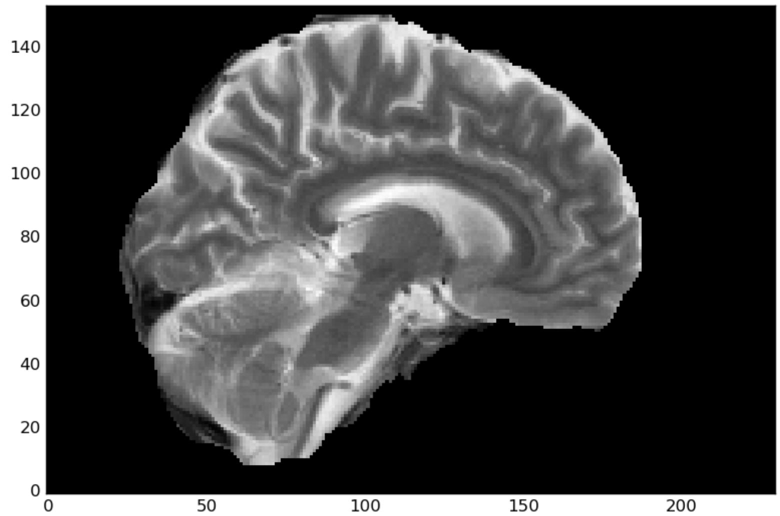


Figure 5.2: Showing a lateral (from side) slice from T2 volume of 23th patient

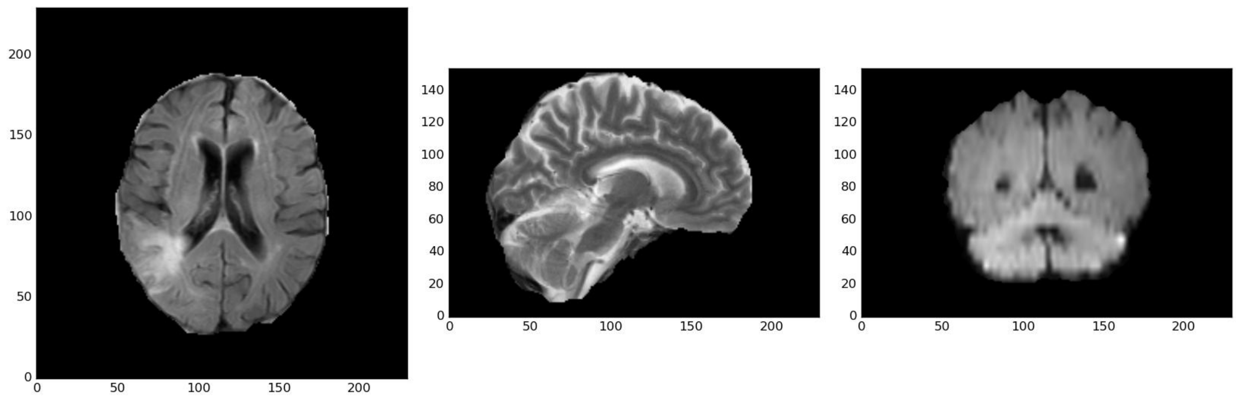


Figure 5.3: Displaying slices from different volumes side by side

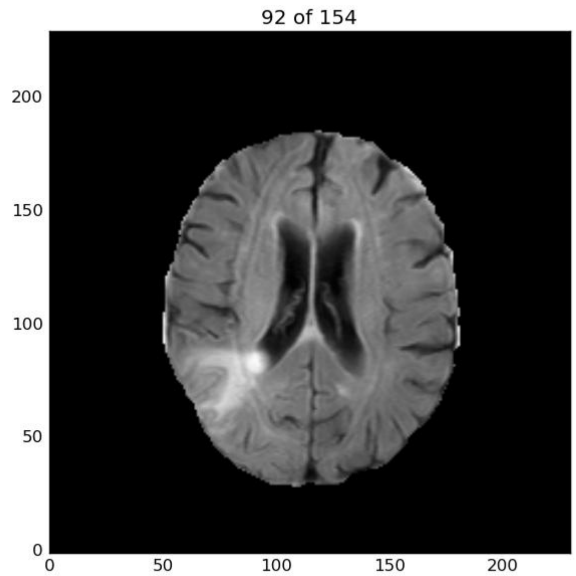


Figure 5.4: 92th frame from the axial animation of Flair data from 11th patient. Notice that lesion area is visible near bottom left of the brain.

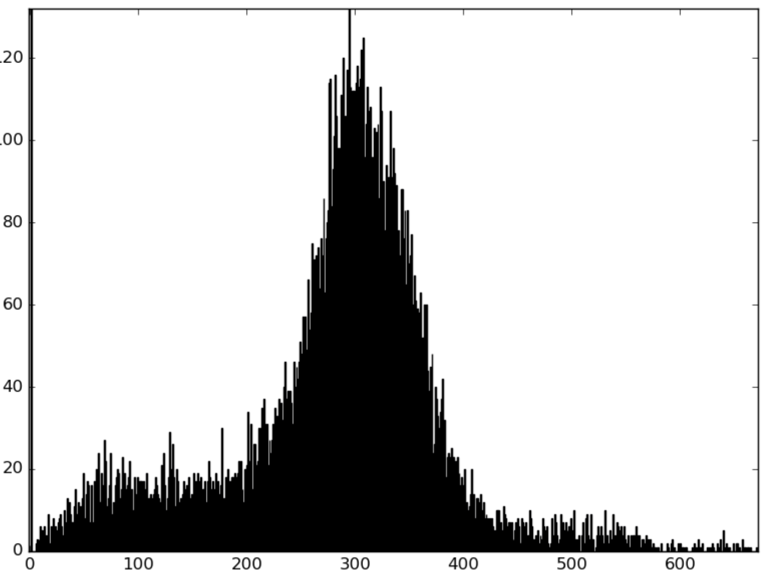


Figure 5.5: Histogram of intensities from the image in the Figure 5.4. Horizontal axis denotes intensity range, vertical axis represents counts of corresponding intensity values. Note that vertical axis is cropped for better visualization, counts of zero intensity values are off the chart.

5.2.2 Implementing The GrowCut Algorithm

GrowCut algorithm is implemented from scratch. The implementation is adapted for three dimensional volume data by performing the algorithm to each two dimensional slice in the volume data. Despite the fact that the algorithm considers 3D data by series of 2D slices, neighborhood function of the algorithm still considers cells in adjacent slices as well as neighboring cells in the same slice.

Another important point to note is as following: the algorithm is limited to work in a specified sub volume near region of interest, because of performance concerns. This limitation is optional and it can be given manually to the corresponding function. Also, seeds should be given manually to the algorithm by inserting several voxel coordinates for both foreground and background regions. An example outcome of this algorithm is given in the Figure 5.6.

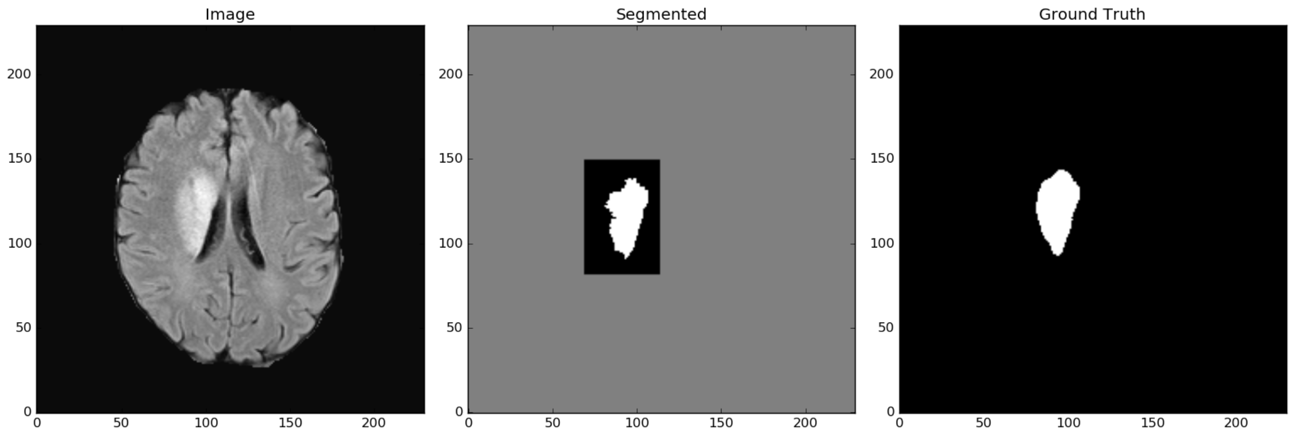


Figure 5.6: An example segmentation result by using GrowCut algorithm for the Flair data of the 15th patient. In the segmentation result, black box indicates limited operation are of algorithm due to performance concerns.

5.2.3 Applying Machine Learning Models

In order to train learning based models, input data are prepared with three dimensional fixed size patches and their labels. Since different MRI volumes for each patient are obtained as pre-registered, one label is used to represent four different MRI modalities. After training, testing procedures are also conducted with patches.

5.2.3.1 Generating Dataset

In the first step of preprocessing data for machine learning models, fixed number of lesion and non-lesion samples are obtained from each modality of each patient.

In this notation, each sample contains four "raw" patches from each modality and one label to represent center voxel of each patch. Also, patch is a cube with NxNxN shape. The keyword "raw" infers patches without any normalization operation such as mean subtraction or standard deviation scaling. Figure 5.7 below explains the notation visually.

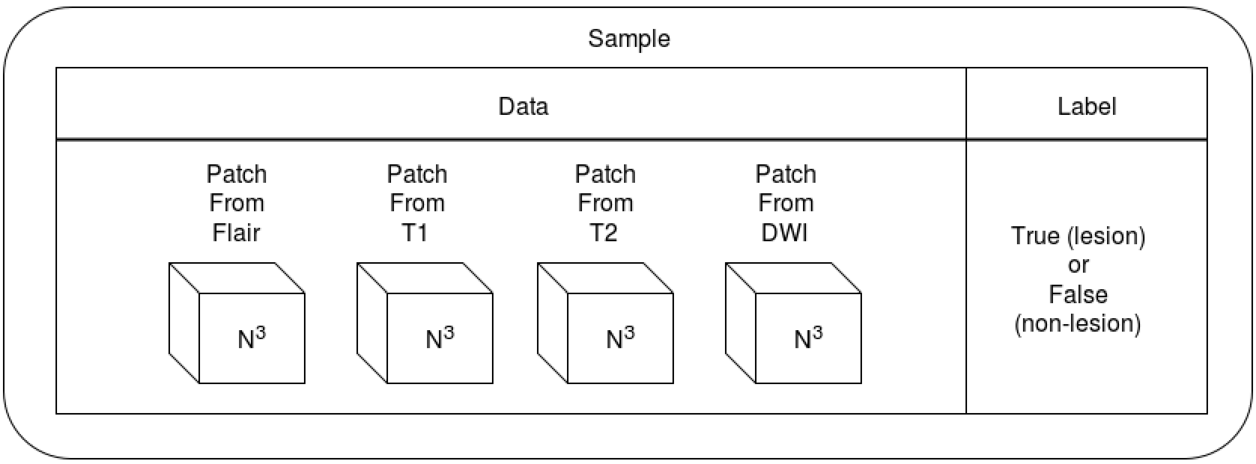


Figure 5.7: Architecture of building block of the dataset

For each patient, 300 lesion and 300 non-lesion samples of four MRI volumes are randomly taken, and patch dimensions are determined as 11^3. Total number of samples per patient is 600, and total count of samples is 15600 since there are 26 patients in all data.

In order to get samples from correct locations, a brain mask is calculated for each patient. Brain mask is a 3D boolean array, where false voxels indicate voxels outside of the brain. After that, lesion and non-lesion masks (3D boolean arrays) are obtained from intersection of brain masks and ground truth data. An example output of this procedure is given below in the Figure 5.8.

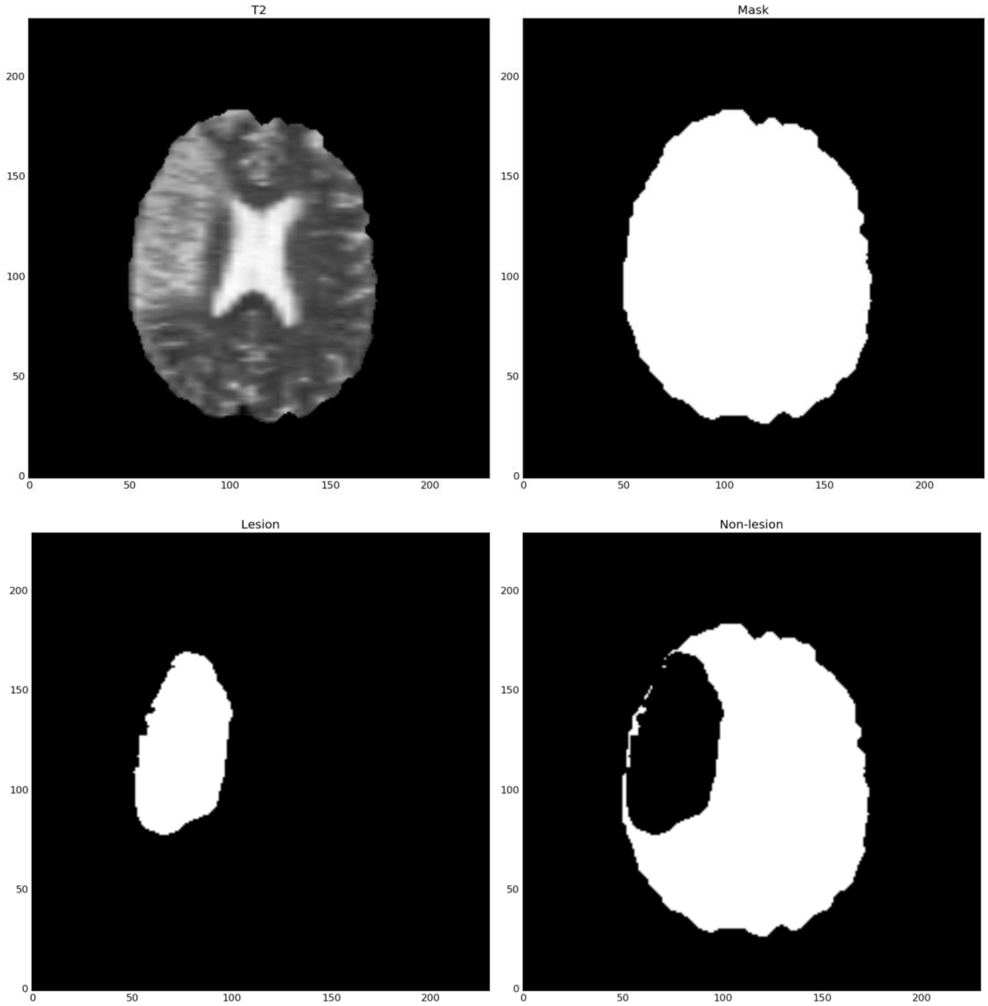


Figure 5.8: Image in the upper left part represents the data to create brain mask. Upper right shows the calculated brain mask. On the bottom left, ground truth of the lesion is displayed. Bottom right shows the locations of tissues that are not lesion.

After obtaining masks for lesion and non-lesion parts, random indices are selected and patches are extracted from these indices. But, it is observed that these random indices infrequently yield empty patches. In order to prevent this condition, patches are checked and random indices are produced repeatedly until acquiring faultless patches.

Pseudocode for creating dataset is given below in Figure 5.9. An example patch is given below in Figure 5.10.

```
procedure PREPAREDATASET
    Initialize dataset ← ∅
    for each patient do
        Load modalities of patient
        Calculate mask_brain of patient
        Apply mask_brain to modalities
        Obtain indices all_lesion and all_non−lesion from
        do
            Get randomIndices from all_lesion and all_non−lesion
            Control randomIndices
        until randomIndices pass control

        for each index in randomIndices do
            Initialize sample ← ∅
            for each modality in modalities do
                Get patch from index and modality
                Add patch to sample
            end for
            Add label of index to sample
            Add sample to dataset
        end for
    end for
end procedure
```

Figure 5.9: Pseudocode for dataset creation.

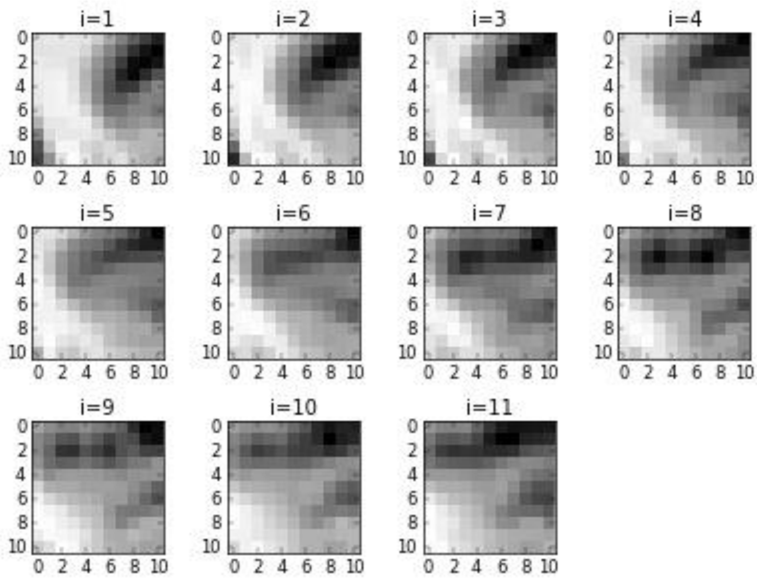
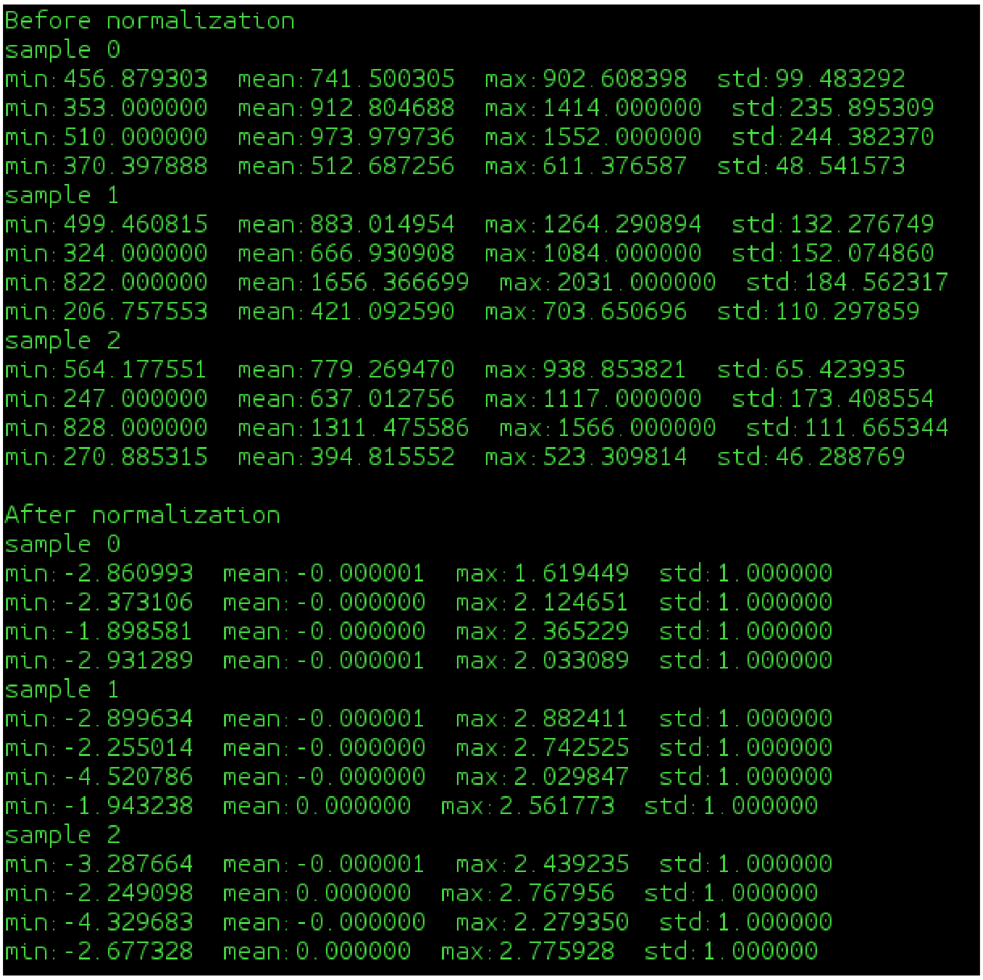


Figure 5.10: An example patch from Flair data of the first patient, labeled as lesion. Label represents the condition of voxel in the middle of the whole patch, which is the center voxel in the sixth slice.

5.2.3.2 Normalizing Dataset

Before utilization of various machine learning models on aforementioned dataset, normalization methods as mean subtraction and standard deviation scaling are applied to each patch in each sample separately. Both of these methods are necessary, since ranges of distinct MRI modalities are not identical. Figure 5.11 shows various metrics from first three samples in dataset, before and after usage of normalization.



```
Before normalization
sample 0
min:456.879303  mean:741.500305  max:902.608398  std:99.483292
min:353.000000  mean:912.804688  max:1414.000000  std:235.895309
min:510.000000  mean:973.979736  max:1552.000000  std:244.382370
min:370.397888  mean:512.687256  max:611.376587  std:48.541573
sample 1
min:499.460815  mean:883.014954  max:1264.290894  std:132.276749
min:324.000000  mean:666.930908  max:1084.000000  std:152.074860
min:822.000000  mean:1656.366699  max:2031.000000  std:184.562317
min:206.757553  mean:421.092590  max:703.650696  std:110.297859
sample 2
min:564.177551  mean:779.269470  max:938.853821  std:65.423935
min:247.000000  mean:637.012756  max:1117.000000  std:173.408554
min:828.000000  mean:1311.475586  max:1566.000000  std:111.665344
min:270.885315  mean:394.815552  max:523.309814  std:46.288769

After normalization
sample 0
min:-2.860993  mean:-0.000001  max:1.619449  std:1.000000
min:-2.373106  mean:-0.000000  max:2.124651  std:1.000000
min:-1.898581  mean:-0.000000  max:2.365229  std:1.000000
min:-2.931289  mean:-0.000001  max:2.033089  std:1.000000
sample 1
min:-2.899634  mean:-0.000001  max:2.882411  std:1.000000
min:-2.255014  mean:-0.000000  max:2.742525  std:1.000000
min:-4.520786  mean:-0.000000  max:2.029847  std:1.000000
min:-1.943238  mean:0.000000  max:2.561773  std:1.000000
sample 2
min:-3.287664  mean:-0.000001  max:2.439235  std:1.000000
min:-2.249098  mean:0.000000  max:2.767956  std:1.000000
min:-4.329683  mean:-0.000000  max:2.279350  std:1.000000
min:-2.677328  mean:0.000000  max:2.775928  std:1.000000
```

Figure 5.11: Statistical metrics of all four patches from the first three samples of dataset, before and after normalization.

5.2.3.3 Classification: K-Nearest Neighbors, Linear, Support Vector Machine and Random Forest

Afterwards normalization of dataset, several algorithms such as K-nearest neighbors classification, linear classification, support vector machine classification and random forest classification are applied via "scikit-learn" package in Python.

Designation of required training and test datasets are performed in two different ways as following: creating datasets by manually selecting training and test sets, shuffling datasets then automatically determining training-test partitions by 5-fold cross validation. Pseudocodes for these approaches are given below in Figure 5.12 and 5.13.

```
procedure evalMlModel
    Initialize model
    Load dataset data, label
    Initialize data_train, label_train, data_test, label_test    ▷ Manually determined
    Train model with data_train, label_train
    Get prediction of data_test
    Calculate confusionMatrix from predictions and label_test
    Calculate metrics from confusionMatrix
end procedure
```

Figure 5.12: Pseudocode for evaluating a model with predetermined data

```
procedure evalMlModelKFoldCV
    Initialize model
    Load dataset data, label
    Initialize cross validation indices as folds
    Initialize prediction_total
    for each fold in folds do
        Initialize data_train, label_train, data_test, label_test    ▷ From fold
        Train model with data_train, label_train
        Get prediction of data_test
        Add prediction to prediction_total
    end for
    Calculate confusionMatrix from prediction_total and label
    Calculate metrics from confusionMatrix
end procedure
```

Figure 5.13: Pseudocode for cross validation technique in model evaluation

Also, it is important to indicate that, patches in each sample are flattened before training of these mentioned models with scikit-learn, thus leading a count of 5324 features (4×11^3) for each sample.

Classification results in terms of sensitivity, specificity and accuracy of these algorithms are given in the section 6.

5.2.3.4 Classification: 3D Convolutional Neural Network

In order to obtain better classification performance, a 3D convolutional neural network model is constructed and applied to the dataset by the instrument of "tflearn" package that provides a simplified interface for "Tensorflow" neural network library in Python.

5.2.3.4.1 Model Details

Network model is constructed with three 3D convolution layers, three 3D max-pooling layers, two fully-connected layers, two dropout layers and one softmax classification layer at the end. Activation functions of 3D convolution layers and fully connected layers are 'leaky_relu'. Also, L2 regularization is applied to all 3D convolution layers.

Input layer consumes data as four-packed 3D volumes. First, second and third 3D convolution layers produce 32, 64 and 128 feature maps respectively. After each 3D convolution layer, one 3D max-pooling layer operates. Next, output of the last pooling layer is given fully-connected layers that have 256 and 512 neurons. Dropout layers are located after each fully-connected layers with 75% keeping probability. Finally, last output is inserted into a layer with two neurons using softmax activation function, where purpose of each neuron is to output a probability value. Architecture illustration of this model is given in Figure 5.14.

5.2.3.4.2 Training

Since the output layer of the model is consist of two layer to indicate probability value of each class (lesion and non-lesion for this case), label of each sample is converted to "one-hot" integer format. In other words, each label is converted to a list where only corresponding index is one, other is zero, for instance label 'False' reorganized as '[1, 0]'.

Then, training of the model is performed with options 25 epoch size and 64 batch size. Evaluation of model is executed as the same way mentioned in previous chapter, manual splitting of training/test dataset and 5-Fold cross validation. Accuracy and loss graphs for the training sessions are supplied in the Figure 5.15 and Figure 5.16 below. Test results are given in the section 6.

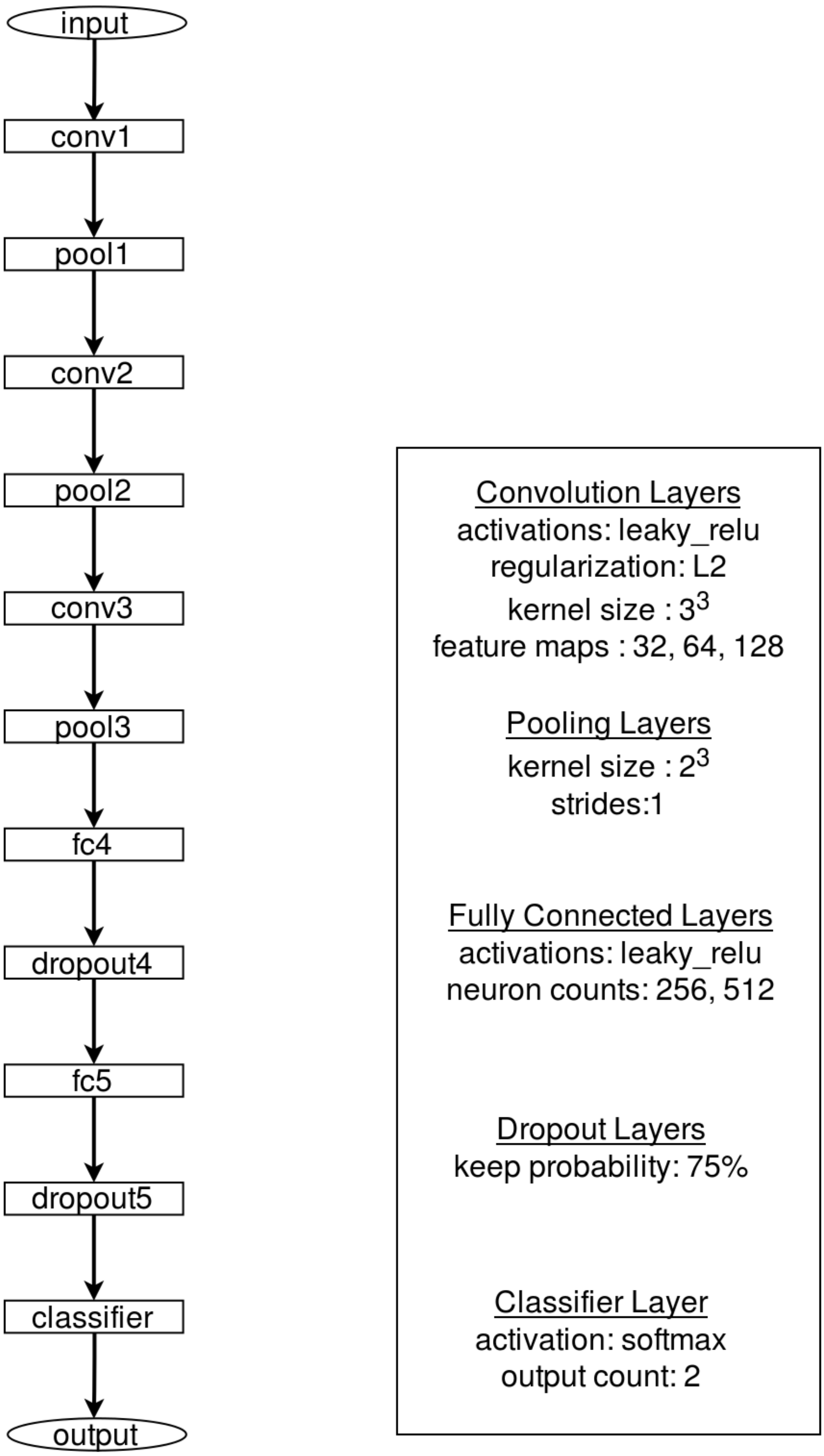


Figure 5.14: Outline of the neural network architecture

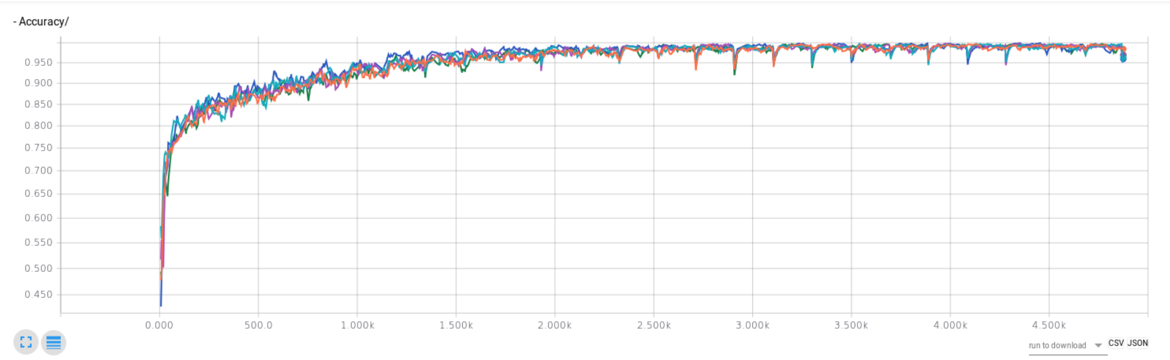


Figure 5.15: Training accuracy graph of all five folds.

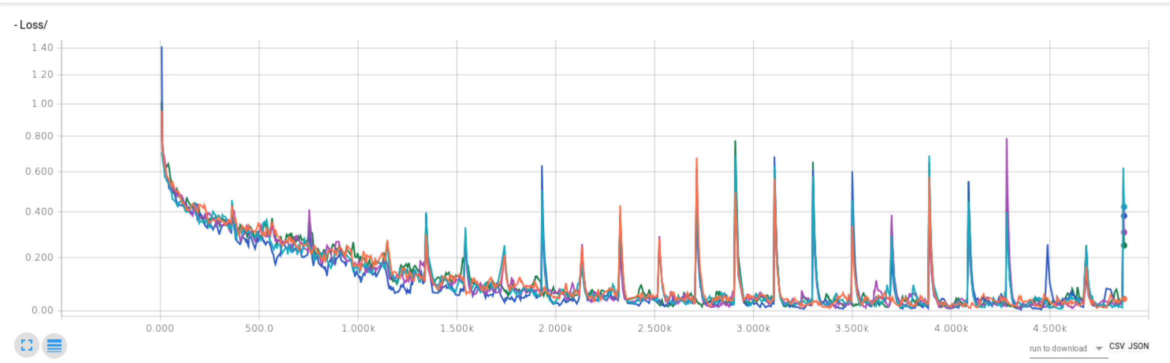


Figure 5.16: Loss graph of the corresponding training session in previous figure.

5.3 Test

Intermediate outputs of the implementation phase are checked simultaneously with implementation, in order to ensure procedures perform in the desired way. This is achieved with the aid of interactive functionalities in IPython development environment.

6. Experimental Results

6.1 Results from GrowCut

After implementation, GrowCut algorithm is tried with comparatively simple cases in terms of size, shape and visibility. Foreground and background seeds are initialized manually. Segmentation results are compared with ground truth values.

Observations from GrowCut segmentation method are as following: number or iterations affect segmentation results dramatically, edges around segmentation results are notably noisy and algorithm is remarkably sensitive to its parameters.

In the below, Figure 6.1 shows results with different iteration counts, when strength parameters of foreground and background are set equally to one. Intuitively, strength of background is increased after this experiment, since the algorithm marked several background voxels as foreground. Result with this new configuration is given in Figure 6.2.

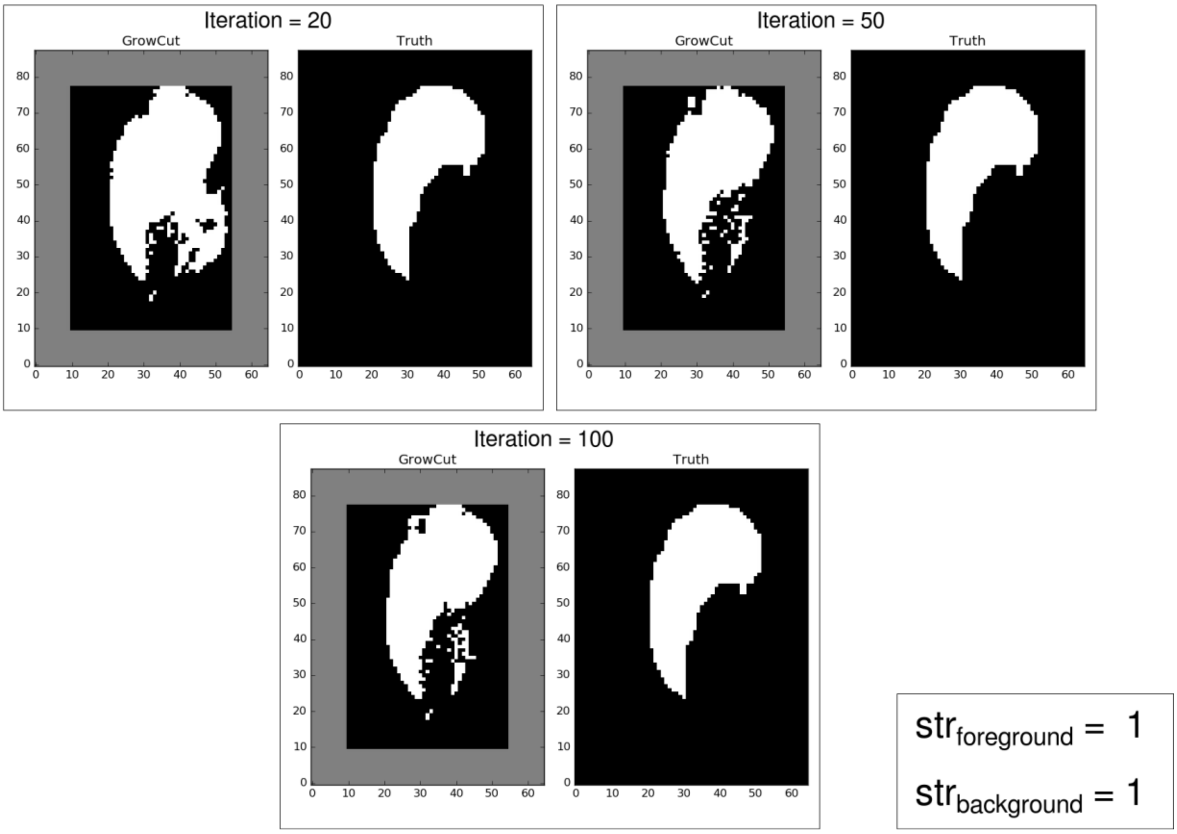


Figure 6.1: Results with ground truth values, from Flair data of 15^{th} patient

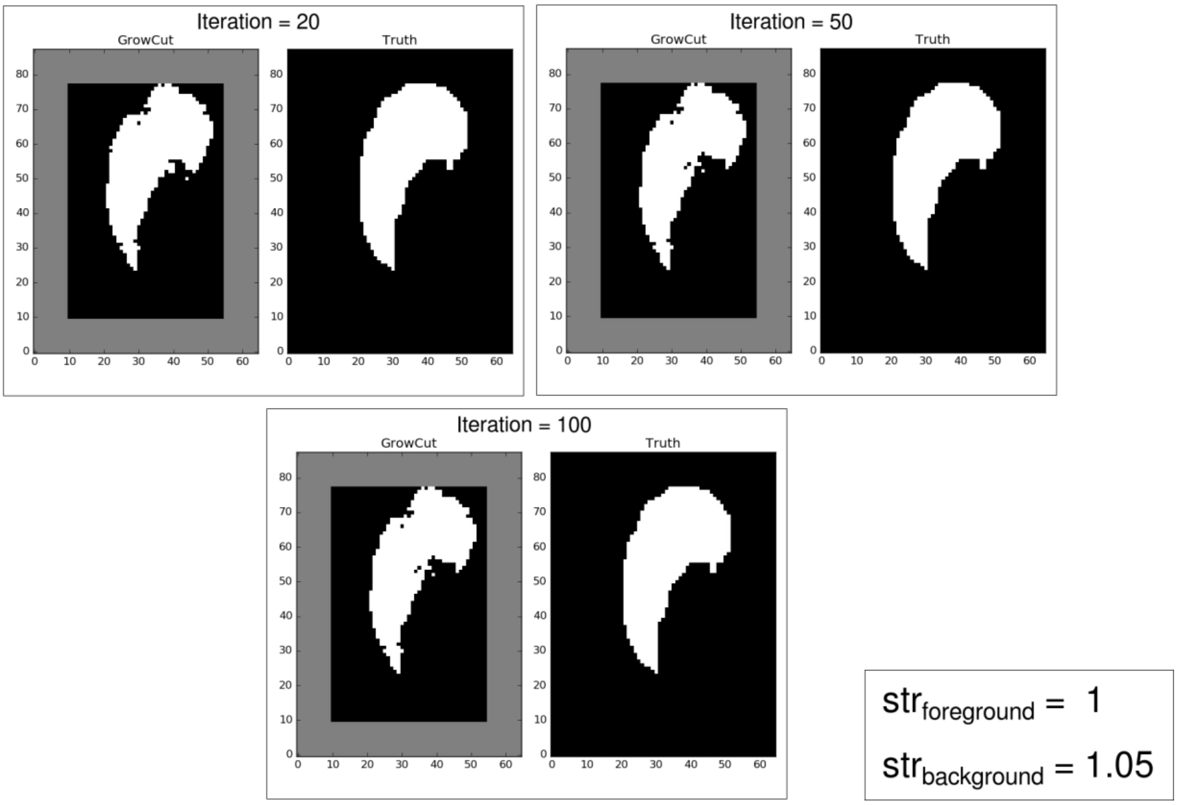


Figure 6.2: Results from same data, after increasing strength of background. Notice that outcomes of iteration 50 and iteration 100 are identical.

6.2 Results from Learning Based Models

Evaluations of machine learning models are conducted with 5-fold cross validation. In order to judge different models fairly, same training and tests sets are used for evaluation of each model. Classification results as counts of true positive, true negative, false positive and false negative are summed through all folds for each model. Then, sensitivity, specificity and accuracy values are calculated.

Table 6.1 below shows these results. It is obvious and that convolutional neural network performs better than other models, since it is considered as the state-of-the-art technology today. However, performance from support vector machine is observed as unexpectedly lesser than other models. Poor parameter choice may have caused this result.

Table 6.1: Patch classification results from learning based models.

Classifier	Confusion Matrix		Sensitivity	Specificity	Accuracy
K-Nearest Neighbors	6891	2116	0,8835	0,7287	0,8061
	909	5684			
Linear Classifier	4760	2006	0,6103	0,7428	0,6765
	3040	5794			
Support Vector Machine	5344	3143	0,6851	0,5971	0,6411
	2456	4657			
Random Forest	5994	1078	0,7685	0,8618	0,8151
	1806	6722			
3D Convolutional Neural Network	7029	942	0,9012	0,8792	0,8902
	771	6858			

7. Conclusion and Suggestions

To sum up, a prominent medical problem, segmentation of ischemic stroke lesions in the brain tissue, is revised from the sight of computer science and engineering.

First of all, problem domain is briefly examined. Definition and nature of the problem are comprehended. Then, literature about solution domain is explored, which consists studies about medical image processing. It is realized that computer vision and machine learning areas suggest various approaches to solve this problem. In the computer vision area, GrowCut algorithm is implemented and tested as a solution candidate. However, GrowCut algorithm requires manual interaction to perform. In order to achieve full automatic segmentation, several machine learning algorithms such as k-nearest neighbors, linear classification, support vector machines, random forests and 3D convolutional neural networks are tried to solve the problem. Finally, results from these techniques are analyzed and compared.

Results of GrowCut algorithm are only examined visually. Algorithm is tested with manual parameter adjustments. It is observed that GrowCut algorithm is able to perform well with proper parameters. Yet, parameters of the algorithm are difficult to fine-tune by hand.

For further improvement, this algorithm can be extended with automatic initialization and automatic adjustment techniques. Also, a formal metric should be utilized to evaluate and improve results, since mere visual inspection is not a decent method for assessment.

On the other side, it is concluded that 3D convolutional neural networks is the best performing technique among machine learning techniques for this problem. Results from machine learning methods are obtained with k-fold cross validation approach and compared with sensitivity, specificity and accuracy metrics. 3D convolutional neural network model performed 89% overall accuracy rate. Second best performing model is random forest classifier, with 81% overall accuracy, and third best model is k-nearest neighbors classifier with 80% accuracy. Linear classifier and support vector machine classifier models accomplished 67% and 64% accuracy respectively. Also note that these results are patch-based classification results.

For additional development about machine learning methods, validation of algorithms can be conducted patient-wise, in other words algorithms can be tested with leave-one-out technique. In this method, all data of a particular patient is excluded from training set. Then, model is trained with that dataset and tested only with the data of excluded patient. This procedure can be applied to all patients respectively. Furthermore, advanced visualization methods, for instance comparison of 3D ground truth and result volumes should aid both assessment and development of learning based algorithms.

Overall, outcome of this project constitutes a feasible solution to the ischemic stroke lesion segmentation problem. It can potentially serve as a starting point to be used in future computer-aided diagnosis and treatment software systems that can aid the physicians in their evaluation of patient outcomes. The methods in this project can be further advanced by means of deeper examination and reformation, additional performance metrics and opinions from medical domain experts.

8. References

[1] National Heart, Lung and Blood Institute, “What Is a Stroke?,” 22-Jun-2016. [Online]. Available: https://www.nhlbi.nih.gov/health/health-topics/topics/stroke. [Accessed: 20-Dec-2016].

[2] World Heart Federation, “Stroke.” [Online]. Available: http://www.world-heart-federation.org/cardiovascular-health/stroke/. [Accessed: 20-Dec-2016].

[3] Daniel J Withey and Zoltan J Koles, “A review of medical image segmentation: Methods and available software,” International Journal of Bioelectromagnetism, vol. 10, no. 3, pp. 125–148, Jan. 2008.

[4] “ISLES: Ischemic Stroke Lesion Segmentation Challenge 2015.” [Online]. Available: http://www.isles-challenge.org/ISLES2015/. [Accessed: 27-Sep-2016].

[5] I. Rekik, S. Allassonnière, T. K. Carpenter, and J. M. Wardlaw, “Medical image analysis methods in MR/CT-imaged acute-subacute ischemic stroke lesion: Segmentation, prediction and insights into dynamic evolution simulation models. A critical appraisal,” NeuroImage: Clinical, vol. 1, no. 1, pp. 164–178, 2012.

[6] J. D. Hunter, “Matplotlib: A 2D graphics environment,” Computing In Science & Engineering, vol. 9, no. 3, pp. 90–95, 2007.

[7] F. Pedregosa et al., “Scikit-learn: Machine Learning in Python,” Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[8] Matthew Brett, ., Michael Hanke, ., Ben Cipollini, ., Marc-Alexandre Côté, ., Chris Markiewicz, ., Stephan Gerhard, ., … Eleftherios Garyfallidis, . (2015). nibabel 2.0.2 [Data set]. Zenodo. http://doi.org/10.5281/zenodo.60846

[9] F. Pérez and B. E. Granger, “IPython: a System for Interactive Scientific Computing,” Computing in Science and Engineering, vol. 9, no. 3, pp. 21–29, May 2007.

[10] E. Jones, T. Oliphant, P. Peterson, and others, SciPy: Open source scientific tools for Python. 2001.

[11] Radiological Society of North America, Inc., “Body MRI - magnetic resonance imaging of the chest, abdomen and pelvis.” [Online]. Available: http://www.radiologyinfo.org/en/info.cfm?pg=bodymr. [Accessed: 20-Dec-2016].

[12] P. Suetens, Fundamentals of medical imaging. Cambridge, UK ; New York: Cambridge University Press, 2002.

[13] David C Preston, MD, “MRI Basics,” Magnetic Resonance Imaging (MRI) of the Brain and Spine: Basics. [Online]. Available: http://casemed.case.edu/clerkships/neurology/Web%20Neurorad/MRI%20Basics.htm [Accessed: 20-Dec-2016].

[14] Neuroimaging Informatics Technology Initiative, “Background Information.” [Online]. Available: https://nifti.nimh.nih.gov/background. [Accessed: 20-Dec-2016].

[15] T S Huang, “Computer Vision: Evolution and Promise.” Journal of Hydraulic Engineering.

[16] Vladimir Vezhnevets and Vadim Konouchine, “‘GrowCut’ - Interactive Multi-Label N-D Image Segmentation By Cellular Automata,” presented at the Proceedings of the 2005 Conference, Graphicon, pp. 150–156.

[17] K. P. Murphy, Machine Learning: A Probabilistic Perspective. Cambridge, MA: MIT Press, 2012.

[18] T. Cover and P. Hart. Nearest neighbor pattern classification. In IEEE Transactions in Information Theory, IT-13, pages 21–27, 1967.

[19] E. Alpaydin, Introduction To Machine Learning, 2nd ed. Cambridge, Mass: MIT Press, 2010.

[20] Andrew Ng, “Support Vector Machines.” CS229 Lecture notes.

[21] I. Goodfellow, Y. Bengio, and A. Courville, “Chapter 6: Deep Feedforward Networks,” in Deep Learning, Cambridge, MA: MIT Press, 2017.

[22] I. Goodfellow, Y. Bengio, and A. Courville, “Chapter 9: Convolutional Networks,” in Deep Learning, Cambridge, MA: MIT Press, 2017.

[23] Michael A. Nielsen, “Chapter 6: Deep Learning,” in Neural Networks and Deep Learning, Determination Press, 2015.

[24] Karpathy et. al., “CS231n Convolutional Neural Networks for Visual Recognition.” [Online]. Available: https://cs231n.github.io/convolutional-networks/. [Accessed: 20-Dec-2016].

[25] “Cross-validation: evaluating estimator performance — scikit-learn 0.18.1 documentation,” 2016. [Online]. Available: http://scikit-learn.org/stable/modules/cross_validation.html. [Accessed: 20-Dec-2016].

[26] K. M. Ting, “Confusion Matrix,” in Encyclopedia of Machine Learning, C. Sammut and G. I. Webb, Eds. Boston, MA: Springer US, 2010, p. 209.

[27] W. Zhu, Z. Nancy, and W. Ning, “Sensitivity, specificity, accuracy, associated confidence interval and ROC analysis with practical SAS® implementations,” Baltimore, Maryland, NESUG proceedings: Health care and life sciences, 2010.

[28] G. van Rossum and F.L. Drake, “6. Modules — Python 2.7.13 documentation.” [Online]. Available: https://docs.python.org/2/tutorial/modules.html. [Accessed: 20-Dec-2016].