Intracranial hemorrhage classification with deep learning

Dominik Klepl
Faculty of Engineering Environment and Computing
Coventry University
Coventry, UK
klepld@uni.coventry.ac.uk

Abstract— Here goes abstract
Keywords— machine learning

I. INTRODUCTION

Intracranial hemorrhage (ICH) is a serious medical condition which if untreated leads to severe disability or death. ICH is usually defined as bleeding occurring inside the skull and can be further categorized based on location of the hemorrhage. These subtypes are:

- Epidural – Located above dura mater which is the first protective layer below skull, wrapping the brain
- Subdural – Located between dura mater and arachnoid which is the second protective layer
- Subarachnoid – Located below arachnoid.
- Intraparenchymal – Located the brain itself
- Intraventricular – Located inside of brain ventricles hollow spaces within brain.

Statistics show that there are 40 000 to 67 000 cases each year in the US. Furthermore, 40% of patients diagnosed with ICH die within 30 days [1] and half of the mortality occurs within first 24 hours [2]. Quick and accurate diagnosis and effective treatment is therefore extremely important in these cases. There are multiple causes of ICH but one of the main ones is stroke which accounts for approximately 15% of the cases [1].

Computed tomography (CT) is the most common tool for ICH diagnosis [2]. CT is preferred because it is usually available in most medical centers and it is a non-invasive method. CT scan can also help with localization and size estimation of the hemorrhage. The diagnostic process in most medical centers is, however, often suboptimal both in its duration and accuracy [3]. The interpretation of CT scans is often done by junior radiologists and is later reviewed by senior staff. Studies show that the initial assessment is often incorrect [4] and as result the necessary treatment might not be delivered in time. An automated diagnostic tool such as the deep learning models reported in this paper might help improve the current diagnostic process both in terms of speed and precision.

A. Working with CT scans

CT scans are not like images such as JPEG or PNG whose pixel values are usually between 0 and 255. CT scans use a standardized image format for storing medical images called DICOM [5]. Output of CT scanner is a 3D image which is rather difficult both to store and view. That is why the 3D images is split into several 2D slices which are then stored as DICOM files.

As mentioned before the scale of pixel values is different from normal images. These values are actually Hounsfield units (HU) with typical range is between -1000 to 3000 [6, 7]. HU is measure of how much the signal sent from the CT machine is attenuated by various materials, in our case blood, bones and various biological tissues. For example, air has -1000 HU, but cortical bone has approximately +1800 HU.

In practice, looking at raw DICOM image is rather useless as the human eye cannot perceive that many color variations. Also, most screens cannot display such range either. When inspecting CT scans, radiologists use various windows [8]. Using a window means to display only limited range of HU, e.g. bone window would show only bones, all other values are set to 0, i.e. black. By going through several windows, radiologist can focus only on relevant parts of the image during diagnosis.

B. Solutions with artificial neural networks

Recently, artificial neural networks and convolutional networks (CNN) in particular proved to be efficient tool in medical image analysis for problems such as classification, object detection and segmentation [9]. Several attempts were also made to train neural networks to classify subtypes of ICH and in some cases even estimate size and location of the hemorrhage. There are two approaches for solving the ICH prediction.

First approach attempts to simulate the workflow of human expert, i.e. radiologist, for ICH diagnosis. Since artificial neural networks are inspired by biological brains it is logical to follow similar path in trying to solve the ICH problem by preprocessing the CT scans to images where human can identify ICH. Usually this is done by applying 3 windows and storing them in color channels of an image. State of the art performance using this approach is reported in [3]. Using convolutional neural networks, they achieved area under the ROC curve of 1.0 for binary classification and 0.91 for subtype classification .

In contrast, second approach does not use CT-specific preprocessing and treats the data as black and white images instead by scaling the CT pixels values to normal pixel scale, e.g. from 0 to 255. Either the raw HU values or values after applying single window are converted [10]. While this approach reduces the complexity of the preprocessing significantly, its performance is lower in comparison to the first approach since much of information is lost due to pixel compression. An example of relatively good results with this approach is [10] who achieved area under ROC curve of 0.846 for binary ICH classification (healthy or any ICH present).

A third approach can be used where instead of optimizing an image for human visual system it is optimized for neural network. To our knowledge, there is no published paper using this approach. It has been proposed rather informally in a Kaggle post [11]. The rationale behind this approach goes as follows. Radiologists use different window settings to explore the image because our visual system can perceive only limited number of color variation, i.e. human eye can differentiate relatively small number of shades of one color in single

picture. However, computers do not have this limitation, so the original pixel values can be used.

The process is quite similar to the second approach but instead of using the raw CT scan we transform the distribution of pixel values to approximately uniform distribution. The raw distribution is usually bi- or multimodal which means that most values are clustered around few values, e.g. 0 and 10. The uniform transformation increases the differences between the pixel values by spreading them across the whole pixel scale. This should help models to "see" the differences in the images better.

In this paper, we compare the performance of models trained using human optimization (i.e. first approach further referred to as HO) and computer optimization (i.e. third approach further referred to as CO) to inspect how the novel approach fares against the state-of-the-art approach.

II. Dataset

The data was published by Radiological Society of North America as part of their competition in collaboration with Kaggle. The dataset comprises of 674258 computed tomography (CT) scans of head stored as DICOM images. These scans come from 61296 patients. Each patient had on average 11 scans. The images were assembled from multiple CT studies done by Stanford University, Thomas Jefferson University, Unity Health Toronto and Universidade Federal de São Paulo. Each image was hand labeled by volunteers from the American Society of Neuroradiology. There are 6 classes: epidural, subdural, subarachnoid, intraparenchymal, intraventricular and any. The last class equals 1 when at least one type of the hemorrhage is present and 0 if no hemorrhage is present. The images can belong to multiple classes, i.e. contain more than one type of hemorrhage. An example of raw image is presented in figure 1.A.

A. Class balance

The number of samples per class differs greatly. There are 577155 images of normal patients, i.e. without any hemorrhage, and 97103 images with hemorrhage(s) present.

In table 1, both the probabilities of each hemorrhage type given a CT scan p(H) and the probabilities given CT scan that contains a hemorrhage p(H|any=1).

In this paper, we deal with the class imbalance in two ways. First, we use undersampling to balance the amount of normal and hemorrhage images. In other words, we reduced the number of normal images to the number as the hemorrhage images by random sampling. After undersampling, the dataset consists of 194082 images from 17066 patients.

Second, weights were introduced to the loss function. The class weights are indirectly proportional to the class sample size so that misclassification of class with small number of images results in higher loss.

III. Data preprocessing

Several preprocessing steps were applied to the images before using either of the image preprocessing approaches described in the introduction (section B). A visual overview of the preprocessing steps is shown in figure X. First, we computed percentage of brain matter in the image and images with less than 2% of brain matter were removed because if almost no brain matter is visible the hemorrhage labels have to be 0. Next step was to zoom in to the head so that the empty space around it is mostly removed. This was done by blurring the image which created a clear demarcation between head and empty space. From this blurred image a mask was created and a square box was drawn around the mask. Finally, the images were cropped to the size of this square box. An example of this zooming and cropping is displayed in figure X.

These partially preprocessed images were further processed using the two approaches, i.e. human optimization and computer optimization.

As described in the introduction, we used two different image preprocessing approaches. The HO approach is filtering the images with three different window settings, brain, bone and subdural, and saving these filtered images as color channels. An example of this preprocessing together with the separate windows is shown in figure X. The CO approach is using the whole range of HU values but transforming their distribution to approximately uniform distribution. An example of this preprocessing is displayed in figure X.

Finally, the images were resized to size 299x299x3 and normalized with min-max normalization so that the pixel values are between 0 and 1.

A. Train-validation-test split

In order to prevent overfitting, we split our dataset into training, validation and testing sets.

10% of data was left out for testing. The remaining 90% were further split into training and validation data with 80% and 10% respectively.

Neither of these splits were random. Two conditions have to be met by each data split. First, the balance of samples from healthy and sick patients was kept at the same level in all data splits, i.e. 1:1. Second, all samples from single patient have to be contained within the same data split. Breaking this condition might cause information spilling between the splits and probably lead to overfitting as the validation and testing data would not be entirely new to the models.

IV. Models

Three artificial-neural-network-based methods were used to predict ICH and its subtypes: convolutional neural network (CNN), transfer learning CNN (TS) and multilayer perceptron (MLP). Each of these methods was used for both image processing approaches. For CNN and MLP we experimented with multiple model architectures but only the best performing model of each method is reported and evaluated in detail.

The general structure of the models is the same for all methods. This is because the ICH classification problem can be essentially divided into two classifications. First of these is a binary classification, no ICH or one or more ICHs. The second one is then classification of ICH subtypes. The models therefore reflect this structure of the problem which makes them multi-output models, i.e. they have two output layers. After some number of layers (dependent on the method) there is the first output layer (further referred to as any-prediction layer) with single neuron and sigmoid activation function which outputs the probability of any ICH present. This output is then concatenated with the output of the layer before the

any prediction layer and passed to the second output layer (further referred to as subtype-prediction layer) with 5 neurons and sigmoid activation function. Sigmoid activation function is used because each image can belong to more than one label.

Furthermore, all models use the Adam optimizer and weighted combination of binary cross-entropy and weighted mean binary cross-entropy as their loss function. Two loss functions are necessary as we have two output layers. Binary cross-entropy as defined in formula (1) was used for the any-prediction layer. The weighted mean binary cross-entropy was used for the subtype-prediction layer. This function is a weighted mean of binary cross-entropies for each ICH subtype. 1.5 times more weight was assigned to the any-prediction loss because it is more important to detect presence of any ICH than it is to know what type(s) of ICH the patient has since the prognosis is similar for all types of ICH.

All models were trained for 30 epochs and batch size of 20. These values were chosen due to limited available computational power After each epoch data were shuffled same to decrease risk of overfitting. The initial learning rate was set to 0.001. If the validation loss was not decreasing for 2 epochs the learning rate was decreased by 50% with the lowest permitted learning rate being $1e^{-8}$. Furthermore, the training could be stopped if the validation loss is not changing significantly for 5 epochs.

A. Convolutional neural network

Convolutional neural networks as class of artificial neural networks that are mostly used for computer vision because the convolutional layers are able to learn to extract features automatically from images. Therefore, there is no need for manual feature extraction which is challenging in case of images.

Features are extracted by the use of many filters in the convolutional layers. At the lowest layers these filters usually become simple edge detectors. The specialization of the filters becomes more abstract the higher the layer is located in the network, e.g. wheel detectors in car classification.

We tested 3 CNN architectures with increasing number of convolution blocks starting from single convolution up to 3 convolution blocks. Each convolution block of our models consists of convolution layer with ReLu activation function, dropout, batch normalization and max pooling. Both dropout and batch normalization should prevent the model from overfitting. Dropout is randomly turning off some neurons in order to prevent the model form fixed signal ways. Batch normalization scales the output of the convolution layer to have mean 0 and standard deviation 1. The max pooling performs down sampling of the number of features that are passed to the following layer by selecting only the features with the highest activation in each region.

An advantage of building CNNs from scratch is that the architecture can be relatively simple and therefore the number of parameters low since the model needs to solve rather narrow-focused problem, i.e. classifying only 1+5 classes. On the other hand, training is computationally expensive and takes a long time.

The best performing architectures of CNN trained on HO and CO images were with X and X convolutional blocks respectively. Their full architectures are reported in appendix X and X for HO and CO.

B. Transfer learning

The second method used in the experiment was CNN with transfer learning (TS). TS means using the convolution part of a previously trained model to extract features and adding a few new layers to perform the classification on top of the network. The weights of the pretrained layers are frozen and only the classification layers are trained. This method assumes that convolution layers trained on sufficiently large and diverse dataset would perform well on new data and labels since the underlying features are the same. In other words, model trained to classify trees from images would be able to extract meaningful features for classifying plants such as shape of leaf because the original model developed leaf detectors.

Architecture of pretrained models used for TS are often complex but capable of extracting wide range of features applicable for many problems. Therefore, they can be more effective than CNNs without TS. Computational cost of training with TS models is relatively low as only a few top layers are trained. On the other hand, the complexity of these models in terms of number of layers and parameters is considerably higher than in case of CNNs without TS. For comparison, the most complex CNN we used has ~83 thousand weights, the TS model has ~22 million weights.

We used the Inception V3 model trained on imagenet dataset [12]. One of the core elements of this model are so called inception layers. Inception layer consists of three parallel convolution layers each with different filter sizes and several max pooling layers. These layers enable the model to choose the size of features that are the most relevant for each image. For example, if there is a small ICH present, the model would use the layer with smaller filters but if the ICH was large the layer with larger filters would be activated. This flexibility in feature extraction is the main reason why we picked this model since the size and shape of ICH can vary significantly and inception architecture should be able to handle this variance efficiently.

C. Multilayer perceptron

Multilayer perceptron (MLP) is one of the simplest neural networks models. The core element of MLP are fully connected layers of neurons.

Unlike CNN, MLP is not capable of feature extraction which poses a challenge if we want to use MLP for image classification. Extracting features manually is complex and requires profound domain knowledge. On the other hand, using single pixels as features is computationally expensive, e.g. with our images of size 299x299x3 this would result in having more than 268 thousand features.

Since we do not have neither the required domain knowledge nor the resources to train MLP using that many features we resized the images to 150x150x3, normalized the pixels and used principal component analysis (PCA) to reduce the number of features. PCA is dimensionality reduction method that can transform data to smaller feature space while preserving as much of variation as possible [13]. This is done by transforming a set of correlated features to uncorrelated vectors i.e. principal components.

Using PCA we reduced the number of features to 50 which were used as input to the MLP models. 5 different architectures of MLP were tested starting from single hidden layer with 50 neurons. Each following architecture added one

hidden layer with number of neurons equal to 75% of neurons of the previous layer.

V. Results

All preprocessing and modelling were done in Python 3.6. Keras with tensorflow backend was used for fitting CNN and TS models. Scikit learn was used for data transformation and fitting MLP.

References

[1] J. J. Heit, M. Iv, and M. Wintermark, "Imaging of intracranial hemorrhage," *Journal of stroke*, vol. 19, no. 1, p. 11, 2017.

[2] J. A. Caceres and J. N. Goldstein, "Intracranial hemorrhage," *Emergency medicine clinics of North America*, vol. 30, no. 3, p. 771, 2012.

[3] H. Ye *et al.*, "Precise diagnosis of intracranial hemorrhage and subtypes using a three-dimensional joint convolutional and recurrent neural network," *European radiology*, pp. 1-11, 2019.

[4] W. Strub, J. Leach, T. Tomsick, and A. Vagal, "Overnight preliminary head CT interpretations provided by residents: locations of misidentified intracranial hemorrhage," *American Journal of Neuroradiology*, vol. 28, no. 9, pp. 1679-1682, 2007.

[5] M. Mustra, K. Delac, and M. Grgic, "Overview of the DICOM standard," in *2008 50th International Symposium ELMAR*, 2008, vol. 1: IEEE, pp. 39-44.

[6] J. Hsieh, "Computed tomography: principles, design, artifacts, and recent advances," 2009: SPIE Bellingham, WA.

[7] M. Gazzaniga and R. B. Ivry, *Cognitive Neuroscience: The Biology of the Mind, Chapter 3*, Fourth International Student Edition ed. WW Norton, 2013.

[8] K. T. Bae *et al.*, "CT depiction of pulmonary emboli: display window settings," *Radiology*, vol. 236, no. 2, pp. 677-684, 2005.

[9] G. Litjens *et al.*, "A survey on deep learning in medical image analysis," *Medical image analysis*, vol. 42, pp. 60-88, 2017.

[10] M. R. Arbabshirani *et al.*, "Advanced machine learning in action: identification of intracranial hemorrhage on computed tomography scans of the head with clinical workflow integration," *npj Digital Medicine*, vol. 1, no. 1, p. 9, 2018.

[11] J. Howard. "DON'T see like a radiologist! (fastai)." Kaggle. https://www.kaggle.com/jhoward/don-t-see-like-a-radiologist-fastai/comments (accessed 23 October, 2019).

[12] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818-2826.

[13] A. Fields, J. Miles, and Z. Fields, "Discovering statistics using R, Chapter 17: Exploratory factor analysis," ed: London: Sage Publications, 2012, pp. 784 - 845.