Article

Deep Learning applied to Intracranial Hemorrhage Detection

Luis Cortes-Ferre [1,†], Miguel Angel Gutiérrez-Naranjo [1,†], Juan José Egea-Guerrero[2,3,†] and Marcin Balcerzyk [4,5,†,*]

1 Department of Computer Sciences and Artificial Intelligence, University of Seville, Sevilla, Spain; luiscortesferre@gmail.com; magutier@us.es
2 Hospital Universitario Virgen del Rocio, Avda. Manuel Siurot, 41013 Sevilla, Spain; juanj.egea.sspa@juntadeandalucia.es
3 Instituto de Biomedicina de Sevilla (Universidad de Sevilla – CSIC – Junta de Andalucía), 41013 Sevilla, Spain; juanj.egea.sspa@juntadeandalucia.es
4 Department of Medical Physiology and Biophysics, University of Seville, Sevilla, Spain; mbalcerzyk@us.es
5 Centro Nacional Aceleradores (Universidad de Sevilla – CSIC – Junta de Andalucía), 41092 Sevilla, Spain; mbalcerzyk@us.es
* Correspondence: mbalcerzyk@us.es; (M.B.)
† These authors contributed equally to this work.

Abstract: Intracranial hemorrhage is a serious health problem requiring rapid and often intensive medical care. Identifying the location and type of any hemorrhage present is a critical step in treating the patient. Diagnosis requires an urgent procedure and the detection of the hemorrhage is a hard and time-consuming process for human experts. In this paper, we propose a novel method based on Deep Learning techniques which can be useful as decision support system. Our proposal is two-folded. On the one hand, the proposed technique classifies slices of computed tomography scans for hemorrhage existence or not, achieving 92.7% accuracy and 0.978 ROC-AUC. On the other hand, our method provides visual explanation to the chosen classification by using the so-called Grad-CAM method.

Keywords: Image Detection; Intracranial Hemorrhage; Deep Learning; Decision Support System.

check for updates

Citation: Cortes-Ferre, L.; Gutiérrez-Naranjo, M.A.; Egea-Guerrero, J.J.; Balcerzyk, M. Title. *Preprints* **2021**, *1*, 0. https://doi.org/

Received:
Accepted:
Published:

1. Introduction

Spontaneous intracranial hemorrhage (ICH) occurs when a diseased blood vessel within the brain bursts, allowing blood to leak inside the brain. The sudden increase of intracranial pressure after bleeding develop brain injury surrounding the blood spill and other cerebral areas. Near of 66 % of all deaths caused by neurological disease world-wide are related to hemorrhagic stroke [1]. According to [2], the overall incidence of spontaneous ICH worldwide is 24.6 per 100,000 person each year and approximately, half of ICH related mortality occurs within the first 24 hours [3].

The severity and outcome of a ICH depends on its cause, brain location, size of the bleed, the amount of time that passes between the bleed and treatment, and so on. ICH usually occurs in selected parts of the brain, including the basal ganglia, cerebellum, brain stem, or cortex. Once brain cells die, they do not regenerate. Damage can be severe and result in physical, mental, and task-based disability. Various types of ICH strike people of all ages. Although ICH are most commonly associated with older adults, they can also occur in younger population. If not treated correctly and immediately, a ICH can lead to disability or death.

In this paper, we present a model based on Deep Learning techniques which can be a useful tool for classifying ICH in both patients and computer tomography (CT) image slices. Our method also includes visual explanations in order to make understandable the decision of the model. We start with a brief introduction, explaining what a intracranial hemorrhage is, and we present the previous state of the art (SOTA) in Artificial Intelligence (AI) methods. Then, we present the network architecture that we use in our experiments.

In addition, as Deep Learning models are usually defined as black boxes since they only provide answers without any explanation or reasoning, we add some grade of explanation to the model decision using the so-called Grad-CAM technique [4]. Grad-CAM provides a heat map that focuses on the pixels that were most influential in the decision.

The paper is organised as follows: In Section 2, some basics on ICH are recalled. In Section 3, some current approaches to the problem of the detection of ICH by Deep Learning and the Grad-CAM techniques are recalled. Section 4, Material and Methods is devoted to explain our proposal. In Section 5, the obtained results are presented. The paper finishes with the discussion of the results (Section 6) and some conclusions. Some appendices have also added with technical details.

2. Intracranial Hemorrhages

In order to understand brain bleeds, it is important to have a basic understanding of the different types based on the location where the bleed occurs (for a detailed introduction to ICH, see, e.g., [5] or [6]). Two types of brain bleeds can occur inside the brain tissue itself - intracerebral hemorrhage (also called cerebral hemorrhage and hemorrhagic stroke) and intraventicular hemorrhage:

- Intracerebral hemorrhage: Bleeding anywhere within the brain tissue itself including the brainstem.
- Intraventricular hemorrhage: bleeding occurs in the brain's ventricles, which are specific areas of the brain (cavities) where cerebrospinal fluid is produced.

Once a stroke has occurred, the cause (bleeding or blood clot) must be determined so that the appropriate treatment can be started. Prompt medical treatment can help limit damage to the brain, which will improve the chance of recovery.

Surgery may be needed since bleeding (hemorrhage) may require immediate decompression of the brain to release pooled blood and relieve pressure. A craniectomy incision (partial removal of the skull to allow the swelling brain to expand), or a craniotomy (opening of the skull cavity).

Depending on the location of the hemorrhage, the extent of damage, the age and the overall health, there can be lasting effects from a brain bleed. These effects can include inability to move part of the body (paralysis), numbness or weakness in part of the body to difficulty speaking or understanding spoken or written words, confusion, memory loss or poor judgment, and so on.

However, over time and with a lot of rehabilitation, it is possible to regain some of these lost functions. Unfortunately, some patients who remain in a coma, or have been severely paralyzed after an intracranial or cerebral hemorrhage may need permanent, long-term care typically provided in a nursing home. Depending on the type, location and extent of the brain bleed, many patients do not survive the initial bleeding event. Mortality rate is as high as 50% in the first 30 days [7].

If there is suspicion of a brain bleed, the sooner the patients get to the emergency room the higher are their chances of survival. Time between the start of symptoms and start of a bleed and between start of a bleed and confirmation of a bleed are critical time points. The earlier a brain hemorrhage is found and classified, the sooner a treatment decision can be made.

3. Deep Learning for the Detection of Intracranial Hemorrhage

Deep Learning [8] is a set of models, techniques and architectures on the basis of Artificial Neural Networks which has represented a revolution on AI in the last years. Its doubtless success in real world-problems reaches areas as face recognition [9], music composition [10] o multilingual translation [11]. In the last decade, thousands of researchers have proposed new models overtaking the achievements of the previous architectures, so it is impossible to give a general framework covering all the possible approaches in Deep Learning. Nevertheless, roughly speaking, a Deep Learning model can be described as a mathematical function which produces an output from a given input. For example, some models provide a person's name if a face is given or a sentence in a language different to the language of the provided sentence. From a technical point of view, such mathematical function is obtained by composition of typically thousands or millions of simple mathematical functions called *neurons* due to their original biological inspiration. These set of neurons is the basis of the neural network and the links among the neurons determine the architecture. The basic architecture corresponds to the so-called multilayer perceptron (MLP, see, e.g., Figure 1), where the neurons are arranged in layers. Each neuron is connected to all neurons in the following layer and there is no connection between neurons of the same layer. As we will see below, from this basic architecture more an more efficient ones can be built. The main feature of such a neural network is that its parameters are self-adjusted by trying to minimize some kind of loss function on a training dataset in a process known as *learning*. After the training process, the neural network with the obtained parameters can be used to solve problems which were unapproachable some years ago.

Deep neural networks can extract a high number of interpretative patterns or features from the data and learn very complex and meaningful representations. Extraction and discovery of these features or patterns can be credited to the depth of the neural network as they are more susceptible to be found at the later layers of the network. The intuition behind it is that these layers progressively learn more complex features. For example, in case of image recognition, the first layer may learn to detect edges, the second layer may learn to identify textures and similarly the third layer can learn to detect objects and so on (see Figure 2). As the nature of the problems being input to the neural network became increasingly difficult, researchers started exploiting deeper and deeper models to achieve better results.

One of the most interesting research areas in Deep Learning is the treatment of digital images, in particular medical images. Among the many application fields, we can cite disease classification [13], ROI segmentation [14] or medical object detection [15]. In this paper, we apply Deep Learning techniques to the study of ICH classification. Recently, some studies have been published on this topic and many researchers have started to pay attention to it. Among many others, we can cite [16], where the authors use a fully convolutional neural network for classification and segmentation with examination of ICH with computed tomographies; [17] where the InceptionV3 and DenseNet Deep Learning models for dealing with CT; or [18], the authors combine Convolutional Neural Networks with other Machine Learning techniques to deal with ICH detection. In [19], the authors use a Dense U-net architecture for detection of ICH. A novel Deep Learning technique based on Monte Carlo algorithm [20] was applied in [21].

The paper Voter *et al.* [22], published in April 2021, deserves special attention. In this paper, the authors have the objective of determining the diagnostic accuracy of an AI decision support system (DSS), developed by Aidoc [23], in diagnosing intracranial hemorrhage (ICH) on noncontrast head CTs and to assess the potential generalizability of an AI DSS. This retrospective study included 3605 consecutive, emergent, adult noncontrast

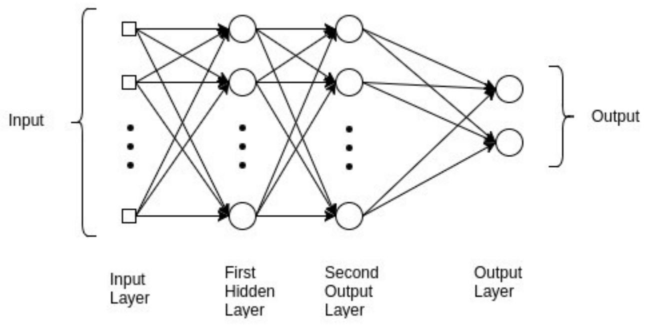


Figure 1. Example of a MLP. Source: A Simple overview of Multilayer Perceptron, Analytics Vidhya.

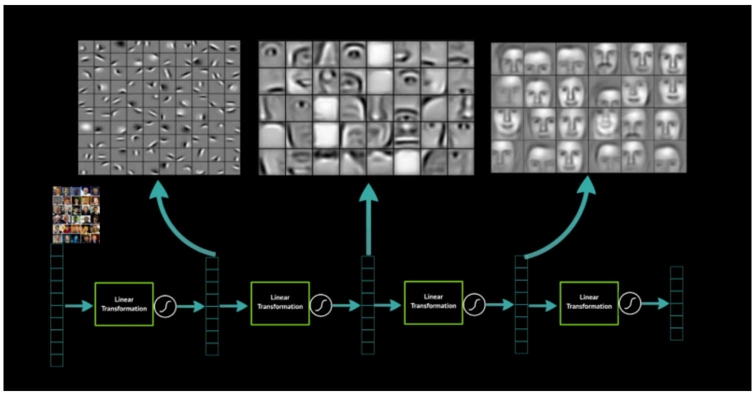


Figure 2. Extraction, learning and integration of features. Source [12].

head CT scans. Each scan was evaluated for ICH by both a certificate of added qualification certified neuroradiologist and Aidoc.

The authors determined the diagnostic accuracy of the AI model and performed a failure mode analysis with quantitative CT radiomic image characterization. The results were: of the 3605 scans, 349 cases of ICH (9.7% of studies) were identified. The neuroradiologist and Aidoc interpretations were concordant in 96.9% of cases and the overall sensitivity, specificity, positive predictive value, and negative predictive value were 92.3%, 97.7%, 81.3%, and 99.2%, respectively.

Aidoc was wrong in 101 of 3605 studies, which corresponds to an accuracy of 97.2%. This study revealed decreased diagnostic accuracy of an AI Decision Support System (DSS) at their institution. Despite of its extensive evaluation, they were unable to identify the source of this discrepancy, raising concerns about the generalizability of these tools. *These results further highlight the need for standardized study design to allow for rigorous and reproducible site-to-site comparison of emerging Deep Learning technologies*.

4. Material and Methods

The aim of this study is two folded. On the one hand, we present a Deep Learning architecture that, in some sense, improves the SOTA methods for classifiying ICH. On the other hand, our method includes visual explanations of the decisions. Tecnically, our proposal combines two of the most successful Deep Learning architectures with a technique for visual explanations of AI models. Next, we provide a short introduction to them.

4.1. Deep Learning architectures

As pointed out above, given a neural network, if we add more layers (more depth), this deeper neural network should have at least the same performance than the shallower one since the bigger neural network would produce the same output than the shallower one if the new layers just learn how to perform like a identity function. However, it was noticed that after some depth, the performance degrades.

This poor performance could be blamed on the optimization function, initialization of the network and, more importantly, the notorious problem of vanishing/exploding gradients, that makes the model unable to learn accurately and efficiently.

In the literature, one can find many attempts to overcome these drawbacks. One of the most successful architectures for solving them is the so-called Residual Neural Networks (or ResNet, for short) [24]. They are inspired in pyramidal cells in the cerebral cortex.

ResNet try to alleviate these problems introducing the so-called *residual blocks*. These blocks introduce a direct connection which skips some layers in between. This connection is called *skip connection* and is the core of residual blocks.

The first basic architecture used in our model is ResNet. The second one is EfficientDet [25]. The main motivation of this architecture is the optimization of resources in Deep Learning models for object detection in digital images. Usually, the big successes obtained with Deep Learning models were obtained with expensive big computers accessible only to big companies. Currently, many efforts try to adapt these models to more simple (and cheap) hardware without losing accuracy. EfficientDet is a clear example of this research line. The main contribution of EfficientDet is two folded. On the one hand, it uses a novel computing unit as the base of the model, the so-called *weighted bi-directional feature pyramid network (BiFPN)* and, on the other hand, the authors propose a compound scaling method for scaling-up several features of the model (resolution, width and depth), that leads to a new family of object detectors called EfficientDet family, from D0 (the smallest one) and D7 (the largest one).

In our study we propose a neural network architecture that combines both EfficientDet [25] and ResNet [24] architectures. Technical details of ResNet and EfficientDet architecture can be found in Appendix A.1.

4.2. Grad-CAM

One of the main drawbacks of the use of Deep Learning techniques in real-world problems is the lack of explainability. In spite of the big accuracy achieved in many cases, it is difficult for the expert to extract the knowledge from the Deep Learning model which justifies the decision. In order to solve this lack of explanations, in the last years, different approached have appeared in the literature. One of the these approaches is the field of visual explanations. Classification models of computational images incorporating such approach usually provide an accurate classification and they also can show the region of the image (the set of pixels) which has been important in the classification process.

One of the methods of visual explanations in Deep Learning is the Gradient-weighted Class Activation Mapping (Grad-CAM) [4] method, where the authors aim to provide visual explanations via gradient-based localization

This approach takes as input an image and it uses the gradients of any output neuron in a classification network flowing into the final convolutional layer to produce a heatmap that highlights the relevant regions in the image to make the corresponding prediction. This method has been also used in [26].

Grad-CAM is a form of post-hoc attention, meaning that it is a method that is applied to an already-trained neural network. The basic idea behind this method is to exploit the spatial information that is preserved through the neural network in order to understand which parts of an input image were important for the classification decision.

Grad-CAM uses the feature maps produced by a convolutional layer (one of the type of layers used in Deep Learning for analyzing data structured in grids, as computational images). The authors argue that the last convolutional layers have the best compromise between high-level semantics and detailed spatial information. The output of Grad-CAM is a class-discriminative localization map, i.e. a heatmap where the hot part corresponds to the part of the image which has been relevant for the model in order to perform the classification. Our model provides two different kinds of information:

- Firstly, our model provides as output if the patient has ICH or not. Technically, the output is the probability of having ICH. The model outputs an affirmative answer if such probability is greater than a fixed threshold.
- Also, our model also provides a color map on the input image where the red area corresponds to the pixels in the image which have been determinant in the decision.

Figure 4 (left) illustrates the use of our model. It shows an image where the model predicts bleeding (ICH) with a probability of 0.9897 and in addition Grad-CAM was able to determine the bleeding area to make that decision. Appendix A.2 provides a more technical description of this technique.

Our code is available online[1]. More details of the framework can be found in Appendix A.3.

[1] https://github.com/Keredu/Intracranial-Hemorrhage-Detection

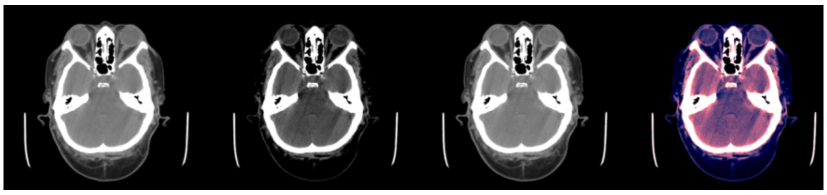


Figure 3. Example of slice windowing as preprocessing of a HU matrix in order to obtain three different matrices (window used between brackets). From left to right: (1) slice with [0,80], (2) slice with [-20, 180], (3) slice with [-150, 230], (4) the three windowed slices stacked (each channel is associated to one color RGB for an visualization purposes.

4.3. Dataset

Kaggle[2] is an online community of data scientists and Machine Learning practitioners that allows users to find and publish data sets, explore and build models in a web-based data-science environment, work with other data scientists and Machine Learning engineers, and enter competitions to solve data science challenges.

The RSNA Intracranial Hemorrhage Competition [27] was a competition hosted by Kaggle at the end of 2019. This competition provides a high number of annotated data (if there is hemorrhage in the slice and the sub-types) separated in two stages. In this work, we only use the data of the second stage (more than 400 GB).

The dataset contains 752,799 scan slices, in Digital Imaging and Communications in Medicine (DICOM) format, from 18,938 patients. These slices provide us matrices with the scan in Hounsfield unit (HU). The HU scale is a linear transformation of the original linear attenuation coefficient measurement into one in which the radiodensity of distilled water at standard pressure and temperature (STP) is defined as zero HU, while the radiodensity of air at STP is defined as -1000 HU. In a voxel with average linear attenuation coefficient μ, the corresponding HU value is therefore given by:

$$HU = 1000 \times \frac{\mu - \mu_{\text{water}}}{\mu_{\text{water}} - \mu_{\text{air}}} \tag{1}$$

where μ_{water} and μ_{air} are respectively the linear attenuation coefficients of water and air.

Thus, a change of one HU represents a change of 0.1% of the attenuation coefficient of water since the attenuation coefficient of air is nearly zero. Therefore, the Eq. 1 is the definition for CT scanners that are calibrated with reference to water.

The set of patients was split into three groups: train (90%, 17044 patients), validation (5%, 947 patients) and testing (5%, 947 patients) groups in order to avoid bias since there are various slices from the same patient. However, it is important to note that there is a remarkable class imbalance since the groups contains the following slices:

- Train: 97525 with ICH, 580934 without ICH.
- Validation: 5401 with ICH, 31174 without ICH.
- Test: 5007 with ICH, 32758 without ICH.

In order to mitigate this class imbalance, we select randomly the same amount of no-ICH slices as the slices with ICH. It turns into 195050 slices from 17044 patients for training, 10802 slices from 947 patients for validation and 10014 slices from 947 patients for testing.

In addition, as our neural network takes three channels as input, the scans with the matrices of the HU must be preprocessed to obtain 3 matrices, one for each channel. In order to obtain such three matrices, three different windows are applied to every slice. Applying a window with X as lower bound and Y as upper bound means that all the HU values lower than X are converted to X, all the HU values greater than Y are converted to Y and the rest remains the same (see Figure 3).

[2] https://www.kaggle.com/

Finally, we build a test dataset with all the slices of the test patients, i.e., 5007 slices with ICH and 32758 without ICH (37765 slices in total). However, in this case we evaluate the model performance annotating every patient with ICH if any slice has ICH and noICH otherwise. Following this annotation criteria, 371 of 947 patients have ICH.

5. Results

The metrics used to evaluate the experiments are the following:

- **Accuracy.** ACC $= \frac{TP+TN}{P+N} = \frac{TP+TN}{TP+TN+FP+FN}$,
- **Sensitivity (Recall).** TPR $= \frac{TP}{P} = \frac{TP}{TP+FN}$,
- **Specificity.** TNR $= \frac{TN}{N} = \frac{TN}{TN+FP}$,
- **Positive Predictive Value (Precision).** PPV $= \frac{TP}{TP+FP}$,
- **Negative Predictive Value.** NPV $= \frac{TN}{TN+FN}$,

where TP = True Positive, FP = False Positive, TN = True Negative, and FN = False Negative.

In Table 1, we present the results obtained from evaluating our model, called EffClass, with the test data of the RSNA Kaggle competition and applying different threshold to our model predictions (i.e., the model predicts ICH if the probability returned by the model is higher than the threshold). The chosen thresholds are 0.5, 0.7 and 0.9. Highest values are marked in **bold**.

We cannot compare our results with the ones shown in the Section 3 since the authors evaluated the images of the patients instead the images slice by slice. Therefore, we evaluate the patients considering the following additional criteria: the patient is diagnosed with hemorrhage if the model predicts, at least, n slices of the image with ICH being n a percentage (5% or 10%) of the number of slices since the number of slices per patient vary. We consider that the model predicts *yes* (slice with ICH) for a certain image slice if the probability is greater than 0.8 to reduce the false positives predictions. Most binary models assume the cutoff value of 0.5, lower than ours.

For example, given two patients with 30 and 40 as number of slices, respectively, and a threshold of 10%, the first one would be classified as patient with ICH if the model predicts that at least three (3) ICH slices but, the second one would be classified as patient with ICH if the model predicts that at least four (4) slices with ICH. The results of the experiments are shown in the Table 2.

Before comparing our results with the results from reference [22], it is very important to remark that we cannot make a fair comparison since the datasets used in our experiments and Aidoc experiments are different. In reference [22] dataset, only the 9.7%, 349 of 3065 patients, were positive (i.e., patients with ICH) meanwhile in our dataset, the 39.2%, 371 of 947 patients were positive. In addition, the results are highly influenced by the types of ICH that compose the dataset and their proportion.

The study presented in [22] shows higher accuracy (ACC) and negative predictive value (NPV). The hghest accuracy value obtained from our experiments is 95.5%, with the EfficientClass model with threshold $=$ 5%. The study [22] shows the same true negative rate (TNR) as EfficientClass with threshold $= 5$. With respect to the true positive rate (TPR), the EfficientClass model with threshold $= 1$ has the higher value (0.965). Finally,

Model (TH)	ACC	TPR	TNR	PPV	NPV	PR	ROC
EffClass (0.5)	**0.927**	**0.914**	0.94	0.938	**0.916**	**0.979**	**0.978**
EffClass (0.7)	0.918	0.872	0.965	0.961	0.883	**0.979**	**0.978**
EffClass (0.9)	0.881	0.774	**0.987**	**0.984**	0.814	**0.979**	**0.978**

Table 1: Results per model and threshold (**TH**). Higher metrics values remarked. **EffClass**: EfficientClassification, **ACC:** Accuracy, **TPR:** Sensitivity (Recall), **TNR:** Specificity, **PPV:** Positive Predictive Value (Precision), **NPV:** Negative Predictive Value, **PR:** Precision-Recall AUC, **ROC:** ROC-AUC.

Model (TH)	ACC	TPR	TNR	PPV	NPV	N	ICH (%)
Aidoc (FDA/501k)	92.9% (*)	93.6% (86.6-97.6)	92.3% (85.4-96.6)	91.7% (84.9-95.6)	94.1% (88.4-97.2)	198	47.5%
Aidoc [22]	97.2% (*)	92.3% (88.9-94.8)	97.7% (97.2-98.2)	81.3% (77.6-84.5)	99.2% (98.8-99.4)	3605	9.7%
EffClass (5%)	95.5% (93.9-96.7)	94.9% (92.1-96.9)	95.8% (93.9-97.3)	93.6% (90.8-95.6)	96.7% (94.9-97.8)	947	39.2%
EffClass (10%)	93.6% (91.8-95.0)	88.1% (84.4-91.3)	97.0% (95.32-98.27)	95.1% (92.3-96.9)	92.7% (90.6-94.4)	947	39.2%

Table 2: Results per model, with at least n% of the slices with ICH. Best metric per model and threshold (**TH**) are marked in bold. **EffClass**: EfficientClassification, **ACC:** Accuracy, **TPR:** Sensitivity (Recall), **TNR:** Specificity, **PPV:** Positive Predictive Value (Precision), **NPV:** Negative Predictive Value, **N:** Number of patients, **ICH (%):** Percentage of patients with ICH. (*): No confidence interval provided.

with respect to the predictive positive value (PPV), the highest value is 0.960, and it was obtained with the EfficientClass model with threshold $= 5$.

As stated in the Section 3, there is a need for standardized study design to allow for rigorous and reproducible site-to-site comparison of emerging Deep Learning technologies.

5.1. Test results obtained with clinical data

Despite the results that we obtained in the previous section are very promising, it needed to test our models with external data to validate that our models could be used not only with Kaggle data but with external data too.

For this purpose, we evaluated the images of five patients from the Hospital Universitario Virgen Macarena [3] (HUVM) and Hospital Universitario Virgen del Rocío[4] (HUVR) patients. Both hospitals are placed in Seville, Spain.

The neural network predictions were all correct (four patients with ICH and only one without ICH) and, in the slices where the probability of IH was very high, we applied GRAD-CAM and we obtained a heatmap with focus in the ICH area. For the Grad-CAM implementation we used the *pytorch-grad-cam* library [28]. In addition, in our Grad-CAM experiments we used augmentation smoothing and, according to the library documentation, it applies a combination of horizontal flips and image multiplications by $[0.9, 1.0, 1.1]$, having the effect of better centering the CAM around the objects.

In Figure 4 a slice with ICH is shown. It is quite difficult to see the ICH but the EfficientClassification neural network predicts ICH with a probability of 0.9897. Figure 5 and Figure 6 show the next two slices where ICH is clearly seen and the neural network predicts ICH with a probability of 0.9972 and 0.9997, respectively. On the other hand, in Figure 7 a slice without IH is shown.

6. Discussion

The irruption of Deep Learning as a novel AI set of new techniques represents a big challenge in medical applications. In many areas, the expertise of the radiologist is the key point to obtain a right diagnostic. In this case, the application to medical image of the latest technology developed for classification of computer images represents a big contribution to human experts. In such way, this paper presents EffClass, a Deep Learning model which

[3] The images from HUVR are provided by courtesy of Dr. Juan José Egea-Guerrero.
[4] The images from HUVM are provided by courtesy of Dra. Soledad Pérez-Sánchez.

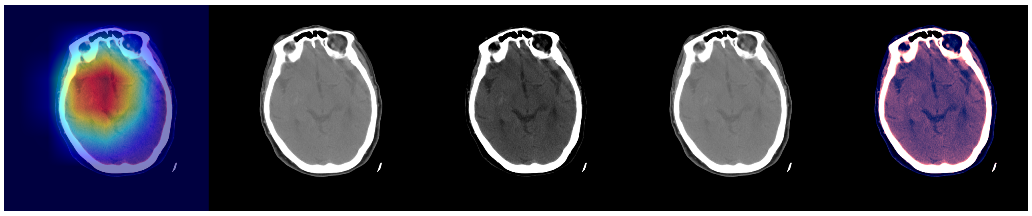


Figure 4. Slice with ICH. The model EfficientClassification predicted bleeding (ICH) with a probability of 0.9897 and in addition Grad-CAM was able to determine the bleeding area to make that decision. From left to right: Grad-CAM, [0,80] window, [-20,180] window, [-150,230] window, the three windows stacked.

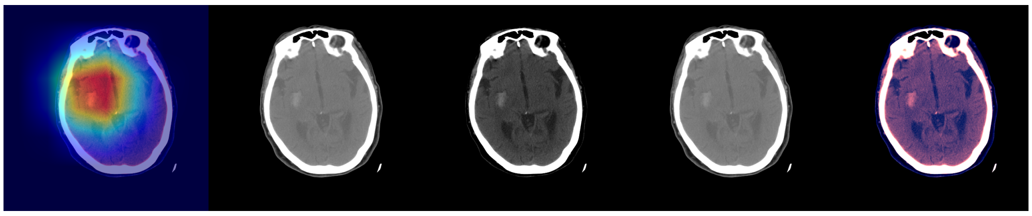


Figure 5. Slice with ICH. The model EfficientClassification predicted bleeding (ICH) with a probability of 0.9972. From left to right: Grad-CAM, [0,80] window, [-20,180] window, [-150,230] window, the three windows stacked.

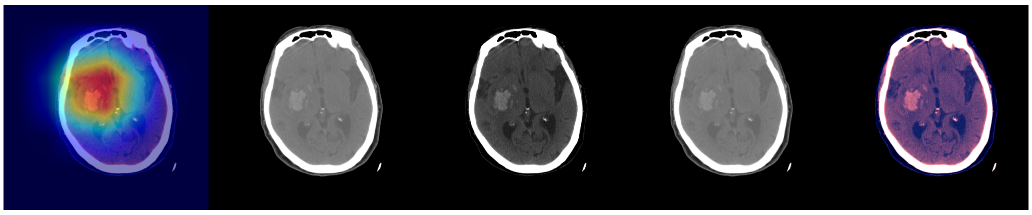


Figure 6. Slice with ICH. The model EfficientClassification predicted bleeding (ICH) with a probability of 0.9997. From left to right: Grad-CAM, [0,80] window, [-20,180] window, [-150,230] window, the three windows stacked.

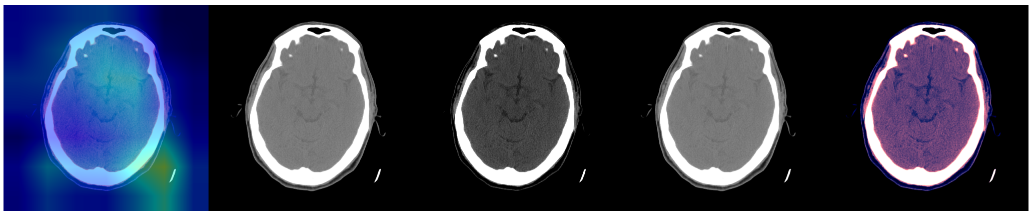


Figure 7. Slice without ICH. The model EfficientClassification predicted bleeding (ICH) with a probability of 0.0097. From left to right: Grad-CAM, [0,80] window, [-20,180] window, [-150,230] window, the three windows stacked.

combines two of the most successful models for image classification and we show that our model is competitive with some of the current most used commercial, CE certified and FDA approved methods like Aidoc [23].

Nevertheless, the use of neural network methods in medicine as support in decision processes has an important drawback. They are usually considered as black boxes where the high accuracy is not taken into account due to the lack of explanations. Human experts not only want to know the decision of the AI system, but the motivation of such decision. In such way, our model not only provides a high accuracy on the decision, but it also provides visual explanation with an intuitive color map on the original image. In such way, Grad-CAM method has been used to highlight the area of the input image that has been relevant in the decision. This mixture of high accuracy together with the visual explanation makes the proposed model as one of the most competitive among the current ones.

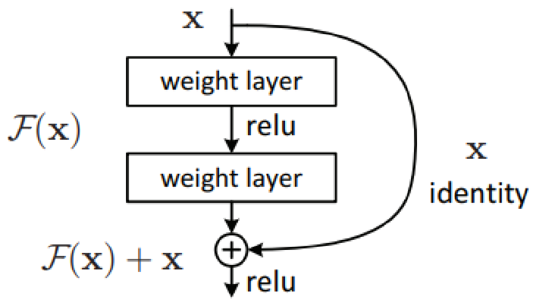

x
weight layer
relu
weight layer
$\mathcal{F}(\mathbf{x})$
$\mathcal{F}(\mathbf{x}) + \mathbf{x}$
relu
x
identity

Figure A1. Residual learning: a building block. Source: ResNet paper [24].

The limitation, as of today, is that the method is for research use only and the command line interface with PyTorch is comfortable only to IT specialists. It requires extraction of the DICOM hospital image from hospital system to a desktop computer. Our method has some higher metrics (ACC, TPR and PPV) than Aidoc FDA submission, bearing in mind that the amplitude of the confidence intervals of Aidoc FDA submission are wide.

Future research lines can be also been considered. From the applications point of view, the Deep learning technologies proposed in this paper can be also applied to obtain classification and visual explanation to other medical decision based on medical images. From a technical point of view, Deep Learning is continuously evolving and other recent technologies on classification and visual explanation deserve to be explored, including 3D approaches.

Author Contributions: All authors contribute equally to this work and have read and agreed to the published version of the manuscript.

Funding: MAGN was funded by Ministerio de Ciencia e Innovación of Spain, grant number PID2019-107339GB-I00. For MB, the study was partially funded by the project CI19-00068 from CaixaImpulse program of La Caixa Foundation (Barcelona, Spain).

Data Availability Statement: Data from Kaggle RSNA Intracranial Hemorrhage Competition are publicly available and can be obtained from [27].

Acknowledgments: We thank Dra. Soledad Pérez-Sánchez for revising the manuscript.

Conflicts of Interest: Juan José Egea-Guerrero and Marcin Balcerzyk are inventors of the patent-pending application WO2019025656.

Appendix A

This appendix include some technical details of our approach.

Appendix A.1

Firstly, some technical details of the models ResNet and EfficientDet are provided. The main idea behind ResNet is to fit a residual mapping instead of hoping each few stacked layers directly fit a desired underlying mapping. Formally, denoting the desired underlying mapping as $H(x)$, ResNet let the stacked nonlinear layers fit another mapping of $F(x) := H(x) - x$. Then, the original mapping is recast into $F(x) + x$. The authors hypothesize that it is easier to optimize the residual mapping than to optimize the original one. For example, if an identity mapping were optimal, it would be easier to push the residual to zero than to fit an identity mapping by a stack of nonlinear layers [24]. A scheme of the so-called *residual blocks* can be found in Figure A1.

If we have two identical architectures but we extend one of them with some layers, we expect to get the same performance at least since, if the network learns to propagate the result of the similar part, both networks should return the same result. However, as the authors stated in the paper, it is not easy to learn the identity function stacking nonlinear layers. With the skip connection, learning the identity function turns to be easier, allowing the networks to learn more than the plain networks.

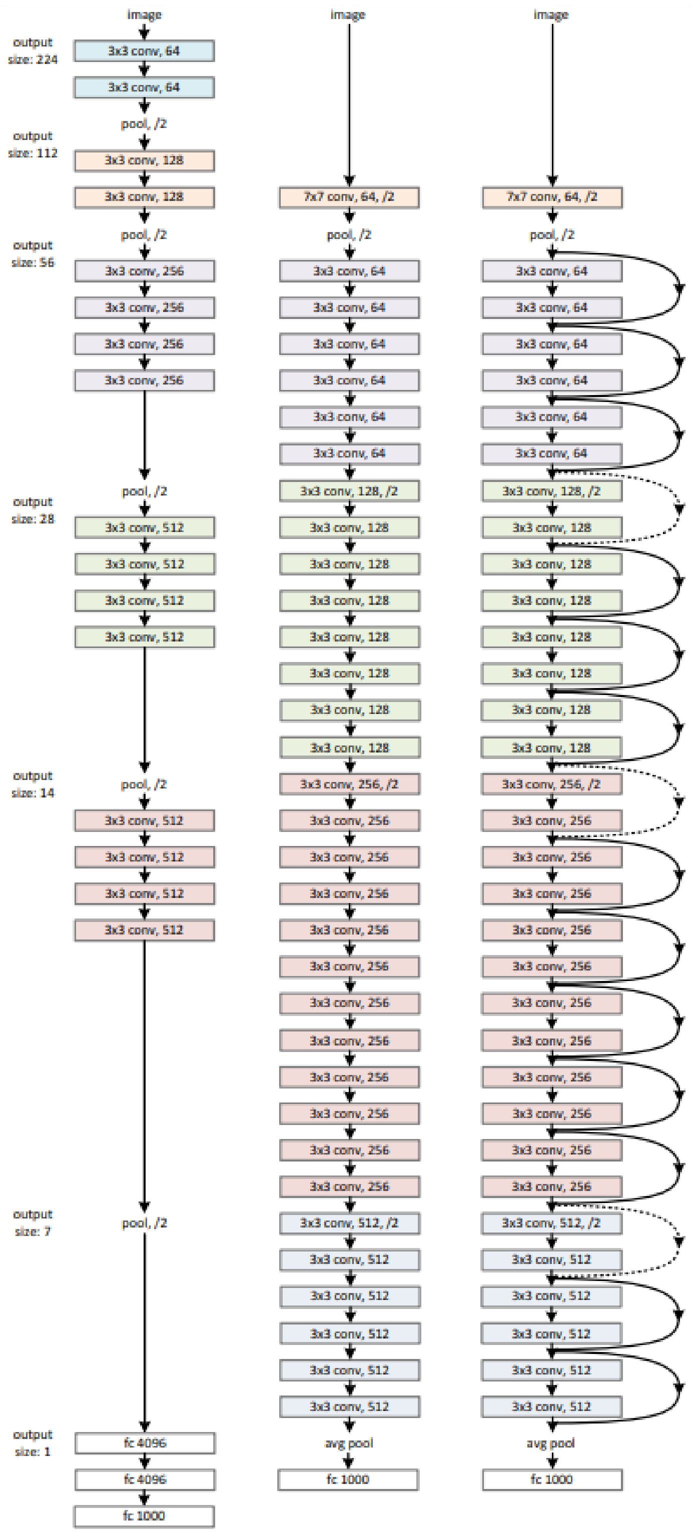


Figure A2. Example network architectures for ImageNet. Left: the VGG-19 model. Middle: a plain network with 34 parameter layers. Right: a residual network with 34 parameter layers. The dotted shortcuts increase dimensions. Source: ResNet paper [24]

In Figure A2, VGG-19 [29], a plain 34-layers network and a ResNet34 neural network are shown. From a technical point of view, our implementation modifies the original architecture of EfficientDet, by means of a substitution of the final layers (the so-called classification and detection heads by a ResNet18 with two units as outputs, representing the probability of ICH.

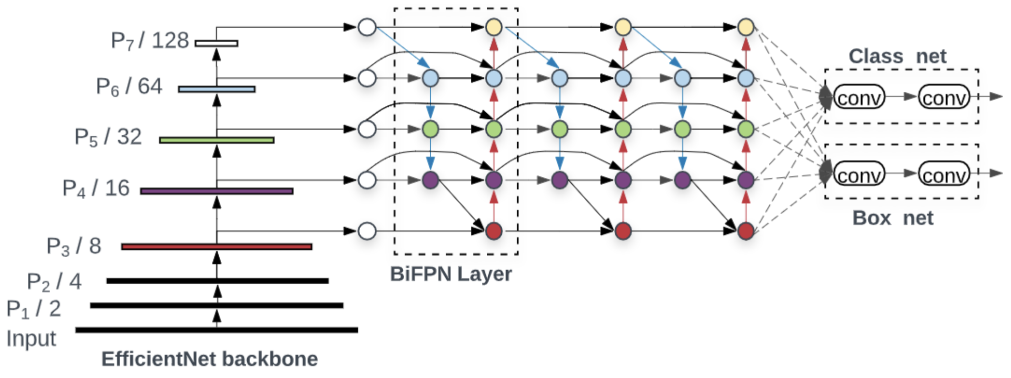


Figure A3. EfficientDet architecture. Source: EfficientDet paper [25].

However, we cannot simply remove the heads and put a fully connected layer instead, since the BiFPN layer provides five sets of feature maps (the five units with colors shown in Figure A3 just before the two heads), each one at different scale: 64 × 64, 32 × 32, 16 × 16, 8 × 8 and 4 × 4, consisting each set of 64 filters. It is possible to flatten the features up and put a fully connected layer with two neurons as output (i.e., create a list with the outputs of the Bi-FPN layer and connect them to two units but this would be probably not enough processing for all these feature maps.

Our approach is to upsample every feature map to 224 × 224, since the ResNet family receives images with a resolution of 224 × 224 as input) and pass it to a ResNet18 but we would have two problems:

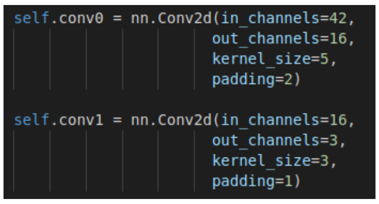
- Upsampling approaches such as upsampling with nearest interpolation lead to the same problem since we are not adding any information.
- Even without adding any information, ResNet would learn from these features but, as ResNet receives 3 × 224 × 224 (Channels, Height, Width) as input, we would need an intermediate step: reduce 320 (5 sets of 64 filters each one) channels to only 3.

. We solve these problems with two intermediate groups of layers between the Bi-FPN and the ResNet: transposed convolution layers as upsampling method and convolution layers to reduce the number of channels.

We call to this neural network **EfficientClassification**. The implementation consists in the following steps:

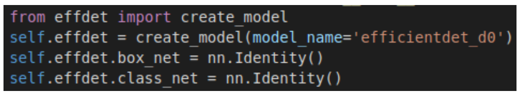
1. **Load EfficientDet.** We get a pretrained EfficientDet-D0. In this step, we use the implementation from the timm library [30]. The timm library can be installed using Pypi.
2. **Cut the regression and classification heads.** We remove the regression and classification heads and only use the EfficientNet backbone and the BiFPN layers. The Figure A5 shows this step and the previous one.
3. **Deconvolution.** The different deconvolution operations (transposed convolutions) are applied to the feature maps to get feature maps of 244 × 224 as dimension. 42 feature maps are passed to the convolution step. The number of channels returned by every deconvolution differs depending on the resolution of the feature maps as can be seen in the Figure A6.
4. **Convolution.** In this step, the number of channels is reduced to 3. Two convolution layers are applied as shown in the Figure A4.

5. **ResNet.** We call a pretrained ResNet18 from torchvision.models and cut the last layer as we did in the ResNet implementation section. The convolution layers output is passed to the ResNet as input.



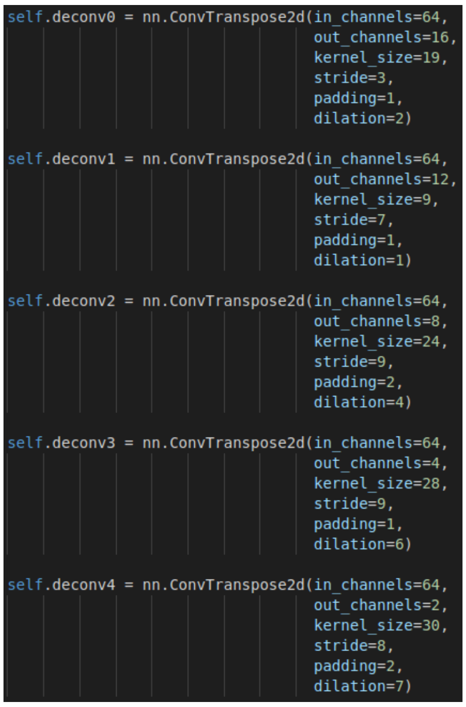
```
self.conv0 = nn.Conv2d(in_channels=42,
                       out_channels=16,
                       kernel_size=5,
                       padding=2)

self.conv1 = nn.Conv2d(in_channels=16,
                       out_channels=3,
                       kernel_size=3,
                       padding=1)
```

Figure A4. Convolution layers. Source: https://github.com/Keredu/Intracranial-Hemorrhage-Detection/blob/main/src/model.py



```
from effdet import create_model
self.effdet = create_model(model_name='efficientdet_d0')
self.effdet.box_net = nn.Identity()
self.effdet.class_net = nn.Identity()
```

Figure A5. Removing the classification and the regression heads from the EfficientDet. Source: https://github.com/Keredu/Intracranial-Hemorrhage-Detection/blob/main/src/model.py



```
self.deconv0 = nn.ConvTranspose2d(in_channels=64,
                                  out_channels=16,
                                  kernel_size=19,
                                  stride=3,
                                  padding=1,
                                  dilation=2)

self.deconv1 = nn.ConvTranspose2d(in_channels=64,
                                  out_channels=12,
                                  kernel_size=9,
                                  stride=7,
                                  padding=1,
                                  dilation=1)

self.deconv2 = nn.ConvTranspose2d(in_channels=64,
                                  out_channels=8,
                                  kernel_size=24,
                                  stride=9,
                                  padding=2,
                                  dilation=4)

self.deconv3 = nn.ConvTranspose2d(in_channels=64,
                                  out_channels=4,
                                  kernel_size=28,
                                  stride=9,
                                  padding=1,
                                  dilation=6)

self.deconv4 = nn.ConvTranspose2d(in_channels=64,
                                  out_channels=2,
                                  kernel_size=30,
                                  stride=8,
                                  padding=2,
                                  dilation=7)
```

Figure A6. Deconvolution layers. Source: https://github.com/Keredu/Intracranial-Hemorrhage-Detection/blob/main/src/model.py

Appendix A.2

Grad-CAM is applied to a trained neural network. Then, the neural network is fed with an image to calculate the Grad-CAM heatmap for that image for a chosen class of interest. Grad-CAM has three steps: compute gradients, calculate alpha values by averaging gradients and calculate the final Grad-CAM heatmap.

Step 1. Compute the gradient of y^c with respect to the feature map activations A^k of a convolutional layer, i.e., $\frac{\partial y^c}{\partial A^k}$, where y^c is the raw output of the neural network for class c, before the softmax is applied to transform the raw score into a probability.

Step 2. Apply global average pooling to the gradients over the width dimension (indexed by i) and the height dimension (indexed by j) to obtain neuron importance weights a_k^c, producing k weights:

$$a_k^c = \frac{1}{Z}\sum_i \sum_j \frac{\partial y^c}{\partial A^k} \tag{A1}$$

Step 3. Perform a weighted combination of the feature map activations A^k with the weights, a_k^c, calculated in the previous step:

$$L^c_{\text{Grad-CAM}} = ReLU\left(\sum_k a_k^c A^k\right) \tag{A2}$$

Tipically, a resize operation is performed afterwards, since the Grad-CAM heatmap size is typically smaller than the original image size.

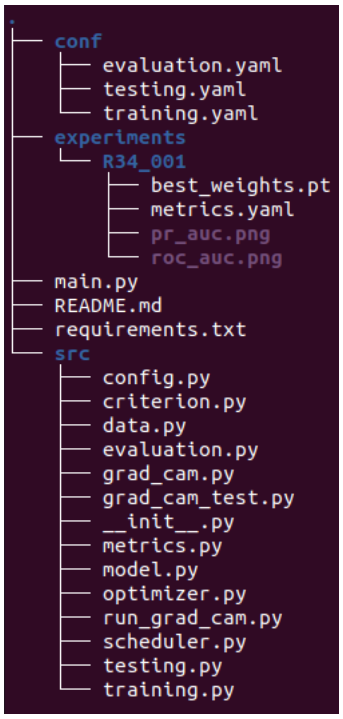
Appendix A.3

Since we needed to conduct a large number of experiments due to both the problem complexity and the network architecture we used, we created a framework that controls the experiments pipeline.

This framework is built on top of the pytorch library [31] and it consists in:

- **Configuration directory**. This directory contains the YAML files which govern the experiments. There are three types of configuration file corresponding to the three different possible phases: training, validation and testing.
- **Experiments directory**. If this directory does not exist, it is created automatically. It contains directories with the experiment code and, inside of them, there are loss and accuracy graphs, the configuration file used in the experiment and the metrics (ROC-AUC, confusion matrix, and so on).
- **Scripts directory**. This directory contains the different scripts that were used in the different phases. The most important one is the **config.py** file, which leads the execution and loads the configuration from the YAML config file.
- **Main file**. The main.py file just receives YAML config files and calls the config.py and the corresponding task script, allowing us to execute multiples experiments.

The Figure A7 shows the framework repository tree a experiment called R34_001.



```
.
├── conf
│   ├── evaluation.yaml
│   ├── testing.yaml
│   └── training.yaml
├── experiments
│   └── R34_001
│       ├── best_weights.pt
│       ├── metrics.yaml
│       ├── pr_auc.png
│       └── roc_auc.png
├── main.py
├── README.md
├── requirements.txt
└── src
    ├── config.py
    ├── criterion.py
    ├── data.py
    ├── evaluation.py
    ├── grad_cam.py
    ├── grad_cam_test.py
    ├── __init__.py
    ├── metrics.py
    ├── model.py
    ├── optimizer.py
    ├── run_grad_cam.py
    ├── scheduler.py
    ├── testing.py
    └── training.py
```

Figure A7. Framework repository tree.

References

1. Valery L Feigin *et al*. Global, regional, and national burden of neurological disorders during 1990–2015: a systematic analysis for the Global Burden of Disease Study 2015. *The Lancet Neurology* **2017**, *16*, 877–897. doi:https://doi.org/10.1016/S1474-4422(17)30299-5.
2. Caceres, J.A.; Goldstein, J.N. Intracranial hemorrhage. *Emerg Med Clin North Am* **2012**, *30*, 771–794.
3. Fogelholm, R.; Murros, K.; Rissanen, A.; Avikainen, S. Long term survival after primary intracerebral haemorrhage: a retrospective population based study. *J Neurol Neurosurg Psychiatry* **2005**, *76*, 1534–1538.
4. Selvaraju, R.R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; Batra, D. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. *International Journal of Computer Vision* **2019**, *128*, 336–359. doi:10.1007/s11263-019-01228-7.
5. Jaja, B.N.; Cusimano, M.D.; Etminan, N.; Hanggi, D.; Hasan, D.; Ilodigwe, D.; Lantigua, H.; Le Roux, P.; Lo, B.; Louffat-Olivares, A.; Mayer, S.; Molyneux, A.; Quinn, A.; Schweizer, T.A.; Schenk, T.; Spears, J.; Todd, M.; Torner, J.; Vergouwen, M.D.; Wong, G.K.; Singh, J.; Macdonald, R.L. Clinical prediction models for aneurysmal subarachnoid hemorrhage: a systematic review. *Neurocrit Care* **2013**, *18*, 143–153.
6. Etminan, N.; Chang, H.S.; Hackenberg, K.; de Rooij, N.K.; Vergouwen, M.D.I.; Rinkel, G.J.E.; Algra, A. Worldwide Incidence of Aneurysmal Subarachnoid Hemorrhage According to Region, Time Period, Blood Pressure, and Smoking Prevalence in the Population: A Systematic Review and Meta-analysis. *JAMA Neurology* **2019**, *76*, 588–597. doi:10.1001/jamaneurol.2019.0006.
7. Al-Kawaz, M.N.; Hanley, D.F.; Ziai, W. Advances in Therapeutic Approaches for Spontaneous Intracerebral Hemorrhage. *Neurotherapeutics* **2020**, *17*, 1757–1767.
8. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press, 2016. http://www.deeplearningbook.org.
9. Fuad, M.T.H.; Fime, A.A.; Sikder, D.; Iftee, M.A.R.; Rabbi, J.; Al-Rakhami, M.S.; Gumaei, A.; Sen, O.; Fuad, M.; Islam, M.N. Recent Advances in Deep Learning Techniques for Face Recognition. *IEEE Access* **2021**, *9*, 99112–99142. doi:10.1109/access.2021.3096136.
10. Hernandez-Olivan, C.; Beltran, J.R. Music Composition with Deep Learning: A Review, 2021, [arXiv:cs.SD/2108.12290].
11. Pan, X.; Wang, M.; Wu, L.; Li, L. Contrastive Learning for Many-to-many Multilingual Neural Machine Translation, 2021, [arXiv:cs.CL/2105.09501].
12. KDnuggets. https://www.kdnuggets.com/2016/02/opening-deep-learning-everyone.html.
13. Shorfuzzaman, M.; Hossain, M.S. MetaCOVID: A Siamese neural network framework with contrastive loss for *n*-shot diagnosis of COVID-19 patients. *Pattern Recognit.* **2021**, *113*, 107700. doi:10.1016/j.patcog.2020.107700.
14. Fan, D.; Zhou, T.; Ji, G.; Zhou, Y.; Chen, G.; Fu, H.; Shen, J.; Shao, L. Inf-Net: Automatic COVID-19 Lung Infection Segmentation From CT Images. *IEEE Trans. Medical Imaging* **2020**, *39*, 2626–2637. doi:10.1109/TMI.2020.2996645.
15. Nair, T.; Precup, D.; Arnold, D.L.; Arbel, T. Exploring uncertainty measures in deep networks for Multiple sclerosis lesion detection and segmentation. *Medical Image Anal.* **2020**, *59*. doi:10.1016/j.media.2019.101557.
16. Kuo, W.; Häne, C.; Mukherjee, P.; Malik, J.; Yuh, E.L. Expert-level detection of acute intracranial hemorrhage on head computed tomography using deep learning. *Proceedings of the National Academy of Sciences* **2019**, *116*, 22737–22745. doi:10.1073/pnas.1908021116.
17. Kishan Das Menon, H.; Janardhan, V. Intracranial hemorrhage detection. *Materials Today: Proceedings* **2021**, *43*, 3706–3714. International Conference on Nanoelectronics, Nanophotonics, Nanomaterials, Nanobioscience & Nanotechnology, doi:https://doi.org/10.1016/j.matpr.2020.10.982.
18. Sage, A.; Badura, P. Intracranial Hemorrhage Detection in Head CT Using Double-Branch Convolutional Neural Network, Support Vector Machine, and Random Forest. *Applied Sciences* **2020**, *10*. doi:10.3390/app10217577.
19. Gruschwitz, P.; Grunz, J.P.; Kuhl, P.J.; Kosmala, A.; Bley, T.A.; Petritsch, B.; Heidenreich, J.F. Performance testing of a novel deep learning algorithm for the detection of intracranial hemorrhage and first trial under clinical conditions. *Neuroscience Informatics* **2021**, *1*, 100005. doi:https://doi.org/10.1016/j.neuri.2021.100005.
20. Kim, J.S.; Cho, Y.; Lim, T.H. Prediction of the Location of the Glottis in Laryngeal Images by Using a Novel Deep-Learning Algorithm. *IEEE Access* **2019**, *7*, 79545–79554. doi:10.1109/ACCESS.2019.2923002.
21. Lee, J.Y.; Kim, J.S.; Kim, T.Y.; Kim, Y.S. Detection and classification of intracranial haemorrhage on CT images using a novel deep-learning algorithm. *Scientific Reports* **2020**, *10*, 20546. doi:10.1038/s41598-020-77441-z.
22. Voter, A.F.; Meram, E.; Garrett, J.W.; Yu, J.J. Diagnostic Accuracy and Failure Mode Analysis of a Deep Learning Algorithm for the Detection of Intracranial Hemorrhage. *J Am Coll Radiol* **2021**.
23. AIDoc Medical Ltd. https://www.aidoc.com/.
24. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. *CoRR* **2015**, *abs/1512.03385*, [1512.03385].
25. Tan, M.; Pang, R.; Le, Q.V. EfficientDet: Scalable and Efficient Object Detection, 2020, [arXiv:cs.CV/1911.09070].
26. Burduja, M.; Ionescu, R.T.; Verga, N. Accurate and Efficient Intracranial Hemorrhage Detection and Subtype Classification in 3D CT Scans with Convolutional and Long Short-Term Memory Neural Networks. *Sensors (Basel)* **2020**, *20*.
27. Kaggle Competition: RSNA intracranial Hemorrhage Detection. https://www.kaggle.com/c/rsna-intracranial-hemorrhage-detection.
28. Gildenblat, J.; contributors. PyTorch library for CAM methods. https://github.com/jacobgil/pytorch-grad-cam, 2021.
29. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *CoRR* **2014**, *abs/1409.1556*.
30. Wightman, R. PyTorch Image Models. https://github.com/rwightman/pytorch-image-models, 2019. doi:10.5281/zenodo.4414861.

31. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; Desmaison, A.; Kopf, A.; Yang, E.; DeVito, Z.; Raison, M.; Tejani, A.; Chilamkurthy, S.; Steiner, B.; Fang, L.; Bai, J.; Chintala, S. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*; Wallach, H.; Larochelle, H.; Beygelzimer, A.; dAlché-Buc, F.; Fox, E.; Garnett, R., Eds.; Curran Associates, Inc., 2019; pp. 8024–8035.