INSTANCE2022 Challenge

Mar Roca - Maxime Tassy - Christophe Avare

Avicenna.AI

Abstract

We designed a very simple 2D Unet-like model and trained it with a binary cross-entropy loss. This configuration achieved a DICE of 0.85 on the test set without post-processing.

Materials and Method

Precise ICH identification and segmentation are difficult tasks in general due to the high variability of presentation depending on the sub-type, time to onset, and many confounding factors like beam hardening or calcifications.

After a first analysis, it seems the dataset contains a majority of intra-axial ICH of different volumes and some extra-axial. In this second category, some contains mixed fluid compartments (a mixture of fresh and old blood) and at least a fully chronic one. Given the dataset size, we do not expect the model to perform equally well on all the sub-types.

All the cases are supposed to be NCCT 5mm, 512x512 matrix. However, the orientation might vary, and there is at least one case which contains a non-image slice.

Given the slice thickness of 5mm, the low number of cases and the ICH volume/sub-type stratification, we decided to use a 2D approach with the trade-off of multiplying the number of individual examples (per slice training) with the loss of potential 3D information.

As a side note, we hypothesize the segmentations have been described with a polygon, with many contours having un-natural sharp shape that would probably not be reproducible by a deep learning approach which tend to extract general patterns, not geometric information. Therefore, we do not expect the network to reach a high DICE or other metric, and we would be happy to target something above 0.8.

Of course, data augmentation is mandatory in a small-data situation, and we performed rotations and mirroring in the axial plane, plus some amount of intensitity shift. Other augmentaiton strategies were not found to have a visible impact neither on performance nor on the final robustness.

The data stratification was based on the presence of a segmentation on a given slice (positive cohort) vs. absence of segmentation (negative cohort). We used during training a balanced 50% / 50% of each cohort per mini-batch.

This category of task is subject to class imbalance as the ratio of positive/negative voxels in a given volume is very low. In an actual project, it is always advisable to minimize this effect by reducing the region of interest to increase this ratio. The best methods generally leverage some anatomically consistent approach, like skull removal and cropping in neuro applications, but given the challenge rules and development time constraints we found developping such approach too risky, so we replaced it with a poor-man's solution by clipping the HU intensities in the soft-tissue range of interest. After some preliminary experiments, we found this approach to work well, probably because it creates large areas of no-contrast that are easily ignored by the model (they do not contain any exploitable information).

At the end of the day, we ended up with a very simple 2D model, less subject to over-learning than larger ones, with a satisfying performance and a straightforward inference pipeline as any 3D volume is just translated into a batch of the size of the axial dimension.

We did not performed any post-processing as we were interested in the raw performance of such model, not the result of some finely tuned or ad-hoc optimization.

Model & Training details

We used the tensoflow 2.9 / Keras framework for training and inference.

Model architecture

We used a simple Unet with a 5 stages encoder/decoder of [8, 16, 32, 48, 64] filters. Each stage is a double [conv/relu/batch norm] block. Concatenation is used for the residual merge opration, and we used a transposed convolution with a stride of 2 in the decoder instead of a bilinear upsampling.

The model input is a layer that performs the clipping operation between [0, 256] and a normalization between [-0.5, 3.5] directly inside the model, so that it will also be performed at inference time:

The model output is the sigmoid of the logits, and the label map is simply the thresholded result of the sigmoid at 0.5.

Training conditions

We first transformed all the examples into two large numpy arrays (resulting from the concatenation along the axial dimension of each example), each one associated to the positive/negative cohort. A generator shuffle the indices and create a mini-batch of the requested ratio of pos/neg examples, then apply the augmentation operations.

The loss function is the binary cross-entropy computed on the logits.

The following hyper-parameters were used:

- Mini batch size: 8
- 5 fold split for training / testing
- Learning rate: 1e-3, with exponential decay of 0.99 after each epoch of 100 steps
- Trained for 10000 steps

Data augmentation was a per-batch (i.e. the batch samples are all augmented or none) random series of the following operations:

- HU shift of [-5, +5] with a 50% probability
- Rotation of [-90, +90] degrees with a 50% probability
