Multi-label classification for CT brain images

wowjenniferlopez

鄭仲堯　鄔世興　張祐祥

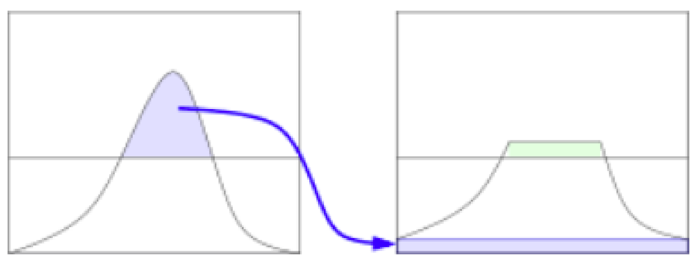
Data augmentation

Dataset Splitting

- We split the patients in the dataset into training and validation parts with the ratio 9 : 1

Contrast Limited Adaptive Histogram Equalization (CLAHE)

- CLAHE limiting the slope of the cumulative distribution function (CDF), which is equivalent to limiting the amplitude of the histogram.
- If the bins in the histogram exceed the upper limit of contrast, the pixels in the histogram will be evenly dispersed into other bins.



Other data augmentation method

- In this project, we also use several common augmentation techniques to avoid training data overfitting
- For instance, flip, rotation, scale and color jittering
- During training and testing, the image data has been resized to 224x224

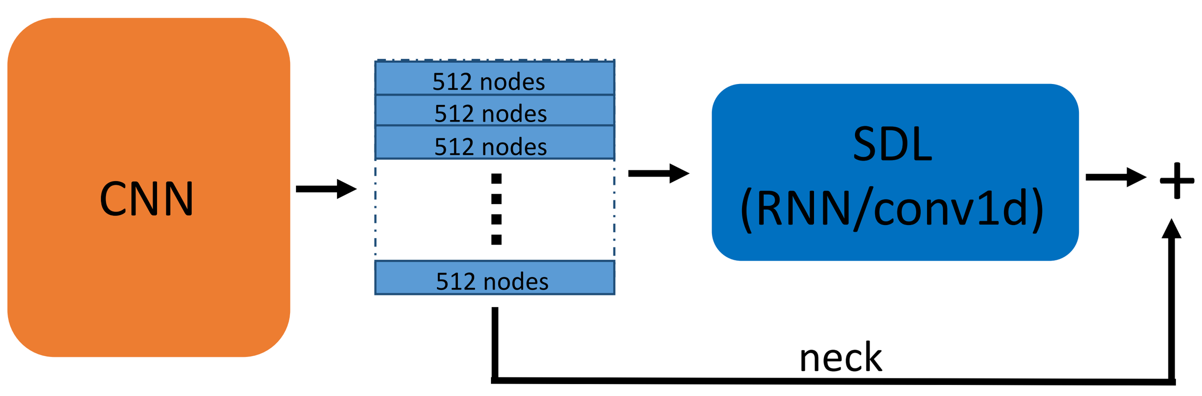
Architecture

CNN backbone

- We use ResNET18 and VGG16 for feature extraction and generates an Nx512 feature matrix
- Due to the limitations of GPU hardware we use VGG16 for the following study

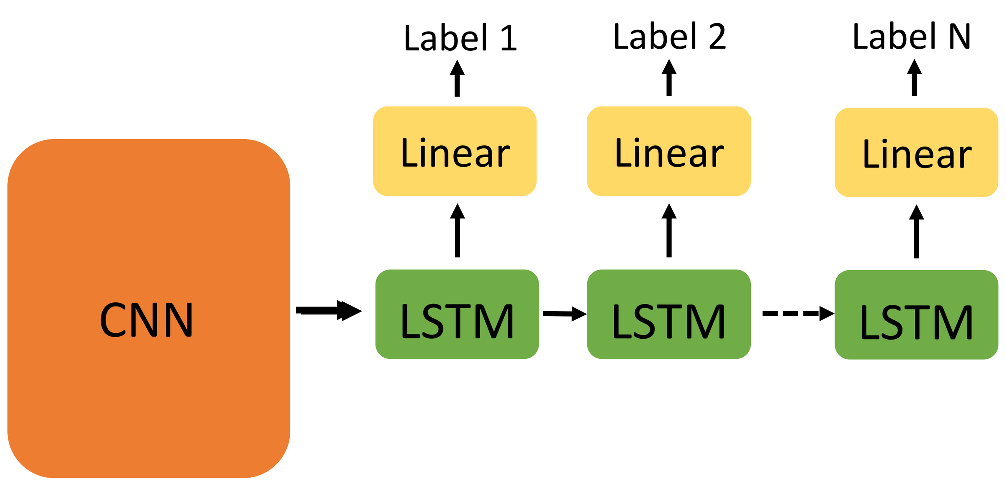
Slice dependencies learning (SDL)

- Intuitively, the relationship between different slices within one patient should be considered
- Here we use RNN/1dconv to learn the dependencies between slices
- We combine CNN output feature as the input of RNN
- Furthermore, we use a "neck", a method that adds RNN output and CNN output to reduce the vanishing gradient problem.

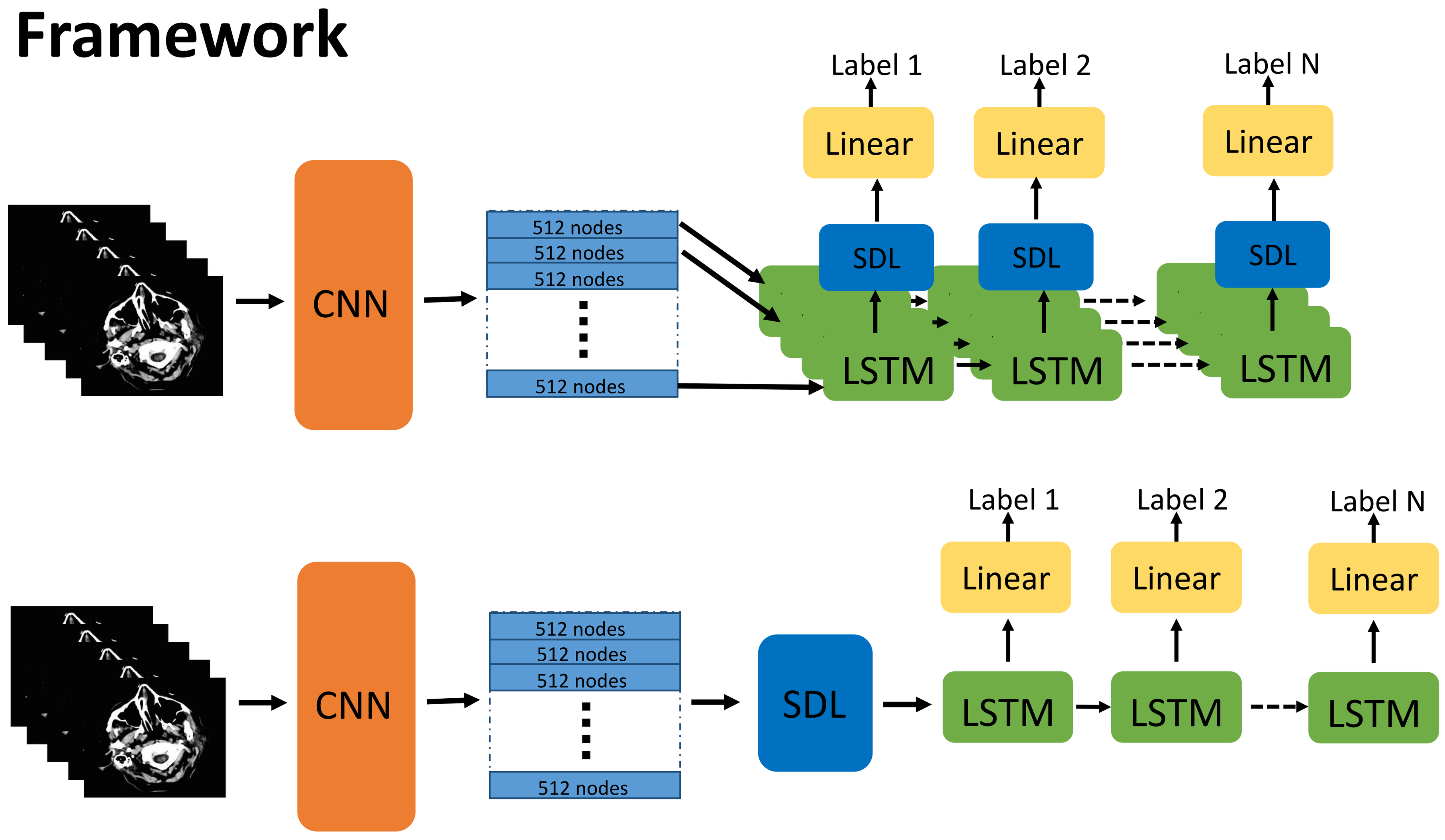


Multi-label classification prediction (LRNN)

- In this part, we further considered the correlation between different trauma or disease
- We use binary decoder RNN to do the final multi-label classification
- This model runs 5 time-steps, each time producing a single scalar which is the score for one particular class, or produce a feature matrix construct by each class
- The CNN feature act as an initial state of LSTM and the input is set to zero



Framework



Training

Asymmetric loss function

- To deal with imbalance data, positive samples is much more important than the negative one.
- ASL is a variation of Binary Cross-Entropy and Focal Loss

$$L = -yL_+ - (1-y)L_-$$

$$\begin{cases} L_+ = (1-p)^{\gamma+} + \log(p) \\ L_- = p^{\gamma-} - \log(1-p) \end{cases}$$

Y : ground-truth label
L_+ / L_- : the positive and negative loss parts
γ : focusing parameter

- ASL as separate focusing parameters r+/r- .
- By setting r->r+ it's easy to emphasize positive samples.

Pretrain CNN backbone

- To reduce training time and increase performance, we pretrain CNN with CT image in the dataset using simple CNN architecture and VGG16 + linear layer with parameters below
  - Input size = 224*224
  - batch size = 16
  - learning rate = 10^{-4}
  - optimizer = Adam

CNN + RNN training

- We use the pretrained VGG16 to set the initial value of the feature extractor in the final model. Using parameters:
  - Input size = 224
  - batch size = not fixed (separate by patients)
  - learning rate = 10^{-5}
  - optimizer = Adam
- We use early stopping technique to choose the best model base on the F2 score, not the accuracy

Experiment result

Model	F2 score (valid)	F2 score (kaggle)
VGG16	0.712	0.721
VGG16 + LRNN	0.720	0.750
VGG16 + SDL(GRU)	0.745	0.768
VGG16 + SDL(1Dconv)	0.747	0.750
VGG16 + SDL(GRU) + LRNN	0.765	0.773
VGG16 + SDL(1Dconv) + LRNN	0.757	0.774
VGG16 + LRNN + SDL(GRU)	0.766	0.783