DLCV Final Project

Zi-Heng Chen
陳子恆
R08922030
Department of Computer Science
r08922030@ntu.edu.tw

Ting-Yu Dai
戴廷宇
R08521605
Department of Civil Engineering
r08521605@ntu.edu.tw

Shao-Yeh Huang
黃紹曄
R08942067
Department of Communication Engineering
r08942067@ntu.edu.tw

Po-Lin Lai
賴柏霖
R08725022
Department of Information Management
r08725022@ntu.edu.tw

1 Model Architecture

Figure 1 shows the architecture of our model. It consists of three components which are backbone, recurrent neural network (RNN) and classfier. We choose ResNet18 as our backbone, a bidirectional gated recurrent unit (bi-GRU) as the second module and use a linear layer to represent the classifier. Suppose x=$\{x_1, x_2, ..., x_m\}$ is a sequence of images of same patient and y=$\{y_1, y_2, ..., y_m\}$ is the corresponding labels. First, x are fed into the backbone and the extracted features g=$\{g_1, g_2, ..., g_m\}$ are fed into RNN. Then, we will obtain h=$\{h_1, h_2, ..., h_m\}$ from the output of RNN. Finally, use the classifier to classify h into L classes and $\hat{y} = \{\hat{y_1}, \hat{y_2}, ..., \hat{y_m}\}$ is our prediction.

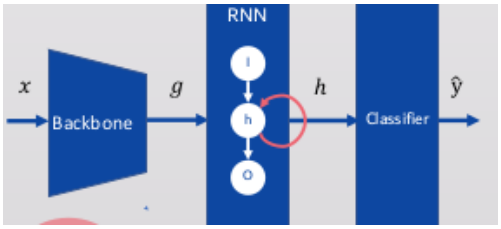

Figure 1: Model Architecture

2 Methodology

Instead of using more complex model, we believe that it is more important to make the model can learn more robust features in this task. Therefore, we train a multitask model (MTM) by introducing different tasks and loss functions to our model.
Suppose $res()$, $rnn()$, $cls()$ are the functions which represent backbone, bi-GRU, classifier, the five different tasks are defined in section 2.1 to 2.5. Section 2.6 shows the loss function of our model. Section 2.7 and 2.8 introduce the training strategies to make the models become more generalizable.

2.1 Multilabel Classification (MC)

This is the main task our work. We need to classify each images into five classes (ICH, IVH, SAH, SDH, EDH). The objective function L_1 of this task is asymmetric loss (ASL)[1].

$$L_1 = -y(1-\hat{y})^{\gamma_+} log(\hat{y}) - (1-y)\hat{y}^{\gamma_-} log(1-\hat{y})$$

ASL enable to dynamically down-weights and hard-thresholds easy negative samples, while also discarding possibly mislabeled samples. ASL is very useful for the positive-negative imbalanced data which is similar in our case.

2.2 Auxiliary Classification (AC)

To make our backbone can learn to extract more robust features, instead of using featuers h, we use the extracted features g and same classifier cls to predict the labels.

$$g = res(x)$$
$$\hat{y}_{AC} = cls(g)$$

The objective function for this task is

$$L_2 = -y(1-\hat{y}_{AC})^{\gamma_+} log(\hat{y}_{AC}) - (1-y)\hat{y}_{AC}^{\gamma_-} log(1-\hat{y}_{AC})$$

2.3 Number of Label Prediction (LP)

Predict the number of labels for each images, this is a single-class classification task and there are six classes in this task (from 0 to 5).

$$h = rnn(res(x))$$
$$\hat{y}_{LP} = cls_2(h)$$

cls_2 is another single linear layer classifier which is different from cls. The objective function for this task is cross entropy loss.

2.4 Distance-based Classification (DC)

This task uses Euclidean distance to measure the correlation between features g_i, $i \in [1, m]$, and label embedding ℓ_j, $j \in [1, 5]$. The distance between g_i and ℓ_j in the same feature space should be small if image x_i contains the label ℓ_j.

$$d_j = f_d(g_i, \ell_j)$$
$$d = [d_1, d_2, ..., d_L]$$
$$\hat{y}_{DC} = -softmax(d)$$

f_d is a function which calculate the similarity between g_i and ℓ_j, such as dot product, cosine similarity, etc. In our case, we use Euclidean distance function as f_d function. The objective function for this task is

$$L_4 = -y(1-\hat{y}_{DC})^{\gamma_+} log(\hat{y}_{DC}) - (1-y)\hat{y}_{DC}^{\gamma_-} log(1-\hat{y}_{DC})$$

2.5 Contrastive Predictive Coding (CPC)

This is an unsupervised learning task[2] which is used to extract useful representations from high-dimensional data. This task aims to predict $g_{t+1}, g_{t+2}, ..., g_{t+T}$ from h_t. Suppose $t = 2$ and $T = 3$, instead of using all sequence g, we will only feed g_1, g_2 into RNN and obtain h_1, h_2, then use a function f_k to calculate the score between h_2 and $g_3, g_4, ..., g_5$.
The objective function of this task is InfoNCE L_5,

$$L_5 = -\mathbb{E}\left[\frac{f_k(g_{t+k}, h_t)}{\Sigma_{g_j} f_k(g_j, h_t)}\right]$$

f_k is a score function which output a score between two vectors. For simplicity, f_k is the dot product function in our case. g_j is the feature vectors from other sequences which are extracted by the backbone. By using InfoNCE, the model aims to get a high score if h and g come from same sequence of images. Otherwise, h and g should obtain a low score if these vectors come from different sequence of images.

2.6 Loss function

The loss function L is defined as

$$L = \alpha L_1 + \beta L_2 + \gamma L_3 + \mu L_4 + \nu L_5$$

2.7 Masking

Before feeding the sequence of feature vectors g into RNN, we will mask some g_i, $i \in [1, m]$. The masking steps are illustrated as follows:

(i) Let $n = m * p_1$

(ii) Choose n images from the sequence g randomly.

(iii) For these chosen g_i, replace them by the zero vectors.

p_1 is the percentage of the masked vectors. After masking the sequence g, we will obtain a masked sequence g_{mask}, then feed g_{mask} into RNN. By using masking strategy, it can help the model become more robust by learning to predict the labels without seeing the whole sequence.

2.8 Diversity Transfer (DT)

There are a limited number of sequences in the training data and it is difficult to make the model become more generalizable by using these sequences only. Therefore, we use a data augmentation technique called "diversity transfer" to generate more sequence for the training data.
Before using this strategy, we need to create a pool which consists of different pairs of images. Each pair of images (z_1, z_2) can be added to the pool if (z_1, z_2) satisfies the following conditions,

(i) z_1 and z_2 are adjacent to each other with z_2 behind z_1.

(ii) z_2 must always have one more label than z_1.

(iii) z_1 and z_2's label sequences must be identical except for the one label z_2 has that z_1 does not. (z_2 contains an additional label ℓ which does not exist in z_1, ex. [0,1,1,0,0] and [0,1,1,0,1])

After creating the pool, the steps of this strategy are shown as follows:

(i) Let $n = m * p_2$

(ii) Choose n images from the sequence g randomly.

(iii) Sample n pairs of images from the pool, then calculate $info(\ell_{k_i})$ for each pair of images. For n pairs of images, we can obtain $\{info(\ell_{k_1})_1, info(\ell_{k_2})_2, ..., info(\ell_{k_n})_n\}$, $k_1, k_2, ..., k_n \in \{1, ..., K\}$ (K = 5 in this task). After doing the subtraction, $info(\ell_{k_i})$ can be treated as the information of label k_i.

$$info(\ell_{k_i}) = res(z_2) - res(z_1)$$

(iv) For i-th chosen images x_i in step (ii), add $info(\ell_{k_i})_i$ to $res(x_i)$ and add ℓ_{k_i} to the labels of i-th chosen images if it does not exist already. So,

$$g_i = res(x_i) + info(\ell_{k_i})_i$$

By using this strategy, we can modify the sequence of feature vectors g and label y. Therefore, the model can learn from more sequences and become more generalizable.

3 Implementation Details

3.1 Data preprocessing

We split the data into training set and validation set. There are 1381 sequences and 45349 images in the training set while the validation set contains 154 sequences and 5143 images.
We use some common augmentation techniques for each image, such as rotation, flipping, change the brightness and contrast, multiscale crop, etc. For each image, we will apply different numbers of augmentation techniques. For example, the first image will be rotated and flipped while the second image will only be cropped.

3.2 Hyperparameter Choices

The batch size is 4 sequences of images. We choose AdamW as the optimizer and the learning rate is 0.001. $p_1 = p_2 = 0.15, T = 8, \epsilon = 0.5, \gamma_+ = 4, \gamma_- = 1$

3.3 Training Stages

If we just train the model by all tasks, masking and diversity transfer strategies, the model cannot be trained successfully. This is due to the loss function and training strategies are too complex. So, we need to pretrain the model first and then fine-tune it afterwards. There are two pretraining stage and one fine-tune stage

(i) In the first pretraining stage, we do not use the masking and diversity transfer strategies and the coefficients of the loss function are defined as follows:

$$\alpha = \beta = 1, \gamma = \mu = \nu = 0$$

(ii) In the second pretraining stage, we use the masking and diversity transfer strategies. The loss function in this stage is as same as the previous pretraining stage.

(iii) After two pretraining stage, we keep using the masking and diversity transfer strategies and fine-tune the model by setting the coefficients as follows:

$$\alpha = \beta = \gamma = \mu = \nu = 1$$

For each sequence of images, we define a probability ϵ to decide whether the model is trained by the original image sequences or the generated image sequence (by applying masking or diversity transfer techniques). For each training stage, we will reduce the learning rate by half each time the validation loss is non-decreasing for three continuous epochs. Each training stage will be stopped if the learning rate is smaller than 10^{-6}.

4 Experiments

We use F2-score to evaluate our performance. Table 1 shows the performance of our model. By using single model, f2-score are 0.78255 and 0.78312 on public and private leaderboard respectively. By using ensemble method, we obtain 0.79385 and 0.79513 on public and private leaderboard respectively. For the small dataset which consists of 307 sequences only, the f2-score on public and private leaderboard are 0.72908 and 0.73567 respectively.

Model	Public Score	Private Score
Single Model	0.78255	0.78312
Ensemble Model	0.79395	0.79513

Table 1: F2-Score on Kaggle Leaderboad

References

[1] Emanuel Ben-Baruch, Tal Ridnik, Nadav Zamir, Asaf Noy, Itamar Friedman, Matan Protter, Lihi Zelnik-Manor (2020) Asymmetric Loss For Multi-Label Classification.

[2] Aaron van den Oord, Yazhe Li, Oriol Vinyals (1995) Representation Learning with Contrastive Predictive Coding.