NeurIPS 2021 BEETL Competition

Benchmarks for EEG Transfer Learning

Michal Nahlik (nahlik.michal@seznam.cz)

Code: https://github.com/michal-nahlik/neurips-beetl-2021

EEG transfer learning

To test the possibility of general EEG transfer learning I implemented multi stage training pipeline and used the same approach in Sleep and Motor imagery task. Model is trained first only on source data and then source data is augmented with target data using mixup method. This allows the model to first learn useful features on source dataset and then adjust it to target data without loosing the learned knowledge. This can unfortunately lead to overfitting so various methods to prevent that are implemented.

Mixup augmentation is a method where two samples are combined together based given mixup rate

$$f(S_1, S_2, r_m) = S_1 * (1 - r_m) + S_2 * r_m \quad ,$$

where S_1 is original sample, S_2 is sample that is going to be added to S_1, and r_m is a mixup rate. Based on the selection method of S_2 we can distinguish two types of mixup, supervised where S_2 is selected to have the same label as S_1 and unsupervised where the label of S_2 is unknown. Being able to adjust mixup rate allows us to ensure how much information of S_1 should be preserved and how much of S_2 information can be inserted.

Training is done in 3 stages:

- Phase 1 (base) - train on source data only
- Phase 2 (mixup) - train on source data with supervised mixup using given target data (samples are selected to have same labels) and higher mixup rate (random for each sample with value up to 0.6)
- Phase 3 (mixup_finetuned) - train on source and target data, source data samples are mixed up in unsupervised way using lower mixup rate (random for each sample with maximum value 0.4)

Using supervised mixup with higher mixup rate will allow the model to learn features from target data correctly for each class, while unsupervised mixup with lower maximum mixup rate transforms the source samples to resemble target data without losing important information needed for correct classification. This approach allows the model to learn, generalize and adjust to target dataset even though the number of labeled target sample is low and to take advantage of unlabeled target data to gain additional knowledge and expand samples available for training. Methods like Virtual Adversarial Training [1][2], label smoothing, random noise augmentation, batch normalization and weight normalization [4] are used to further lower the chances of overfitting and help model generalization.

Common settings of training pipeline (for both tasks):

- Optimizer: AdamW (weight_decay=5e-4)
- Loss: Focal loss [3] with label smoothing, Virtual Adversarial Training [1][2] (implementation from: https://github.com/lyakaap/VAT-pytorch)
- Augmentations: Random noise
- Weight normalization of linear layers [4]
- Data normalization using standard scaling
- Trained two models with different seeds (42, 2021), average output is used as final prediction

Sleep task

Data preprocessing: standard scaling base on mean and standard deviation calculated over all values in source dataset. These values are then used to scale also target and test samples.

Model: inspired by Multi-Resolution CNN proposed by Eldele et al 2021 [5], used with modifications to activation function, number of layers, kernel sizes etc.

Dataset setup:

- Source dataset: source dataset
- Target dataset: target data from leaderboard testing phase and final phase
- Unsupervised mixup data: final phase test data

Training setup:

- Phase 1 – 5 epochs, OneCycleLR scheduler, learning rate 0.001
- Phase 2 – 10 epochs, no scheduler, learning rate 0.0001, supervised mixup using target data with maximum mixup rate 0.6
- Phase 3 – 20 epochs, OneCycleLR scheduler, learning rate 0.001, unsupervised mixup using test data with maximup mixup rate 0.4, WeightedRandomSampler with 0.75 sample weight on target samples and 0.25 for source samples and sample rate 0.5

Motor imagery task

External source datasets: Cho2017 (subjects 1-8), PhysionetMI (subjects 1-14), BNCI2014001 (subjects 1-8)

Data preprocessing: resampled to 100 Hz, band-pass filter for frequencies 4 – 30 Hz, standard scale each session using mean and standart deviation from all samples in the session, use only common channels (17), relabel to 3 classes

Model: improved base EEGShallowClassifier[6] with attention module proposed by Liu et al 2020[7] (implementation from: https://github.com/Shenyonglong/Spatial-Temporal-attention-/tree/master/models), added batch normalization after every convolution, increased number of filters (n_filters_time=64, n_filters_spat=64) and replaced EEGClassifier by Linear layer with weight normalization

Dataset setup:

- Source dataset: external source datasets, target dataset A and B from leaderboard phase
- Target dataset: final target dataset A and B
- Unsupervised mixup data: leaderboard and final test data for dataset A and B

Training setup:

- Phase 1 – 10 epochs, OneCycleLR scheduler, learning rate 0.001, WeightedRandomSampler with 0.75 sample weight on leaderboard target samples and 0.25 for external source samples
- Phase 2 – 10 epochs, no scheduler, learning rate 0.001, supervised mixup using target data with maximum mixup rate 0.6
- Phase 3 – 20 epochs, OneCycleLR scheduler, learning rate 0.001, unsupervised mixup using test data with maximup mixup rate 0.4, WeightedRandomSampler with 0.75 sample weight on target samples and 0.25 for source samples and sample rate 0.5

Discussion

I implemented several methods for transfer and semi-supervised learning including Mean Teacher[8] and Deep Adaptation Networks[9] but proposed approach of multistage training with mixup and Virtual Adversarial Training [1][2] provided best results in both tasks. I focused mainly on Sleep task where I was able to get a good base solution even without transfer learning and proposed method improved the results significantly. My base results in Motor imagery were not as good but still proposed training method gave a decent improvement. Another important parts of the solution are standard scaling of samples and heavy use of batch normalization which allows the model to learn even on samples from different datasets. I tested also other preprocessing methods like robust scaler or quantile transformation but standard scaler lead to the best results.

Bibliography

[1] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, Ken Nakae, & Shin Ishii. (2016). Distributional Smoothing with Virtual Adversarial Training.

[2] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, & Shin Ishii. (2018). Virtual Adversarial Training: A Regularization Method for Supervised and Semi-Supervised Learning.

[3] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, & Piotr Dollár. (2018). Focal Loss for Dense Object Detection.

[4] Tim Salimans, & Diederik P. Kingma. (2016). Weight Normalization: A Simple Reparameterization to Accelerate Training of Deep Neural Networks.

[5] Eldele, E., Chen, Z., Liu, C., Wu, M., Kwoh, C.K., Li, X., & Guan, C. (2021). An Attention-Based Deep Learning Approach for Sleep Stage Classification With Single-Channel EEG. *IEEE Transactions on Neural Systems and Rehabilitation Engineering, 29, 809-818.*

[6] Schirrmeister, R., Springenberg, J., Fiederer, L., Glasstetter, M., Eggensperger, K., Tangermann, M., Hutter, F., Burgard, W., & Ball, T. (2017). Deep learning with convolutional neural networks for EEG decoding and visualization. *Human Brain Mapping, 38(11), 5391–5420.*

[7] Liu, X., Shen, Y., Liu, J., Yang, J., Xiong, P., & Lin, F. (2020). Parallel Spatial–Temporal Self-Attention CNN-Based Motor Imagery Classification for BCI. *Frontiers in Neuroscience, 14, 1157.*

[8] Antti Tarvainen, & Harri Valpola. (2018). Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results.

[9] Mingsheng Long, Yue Cao, Jianmin Wang, & Michael I. Jordan. (2015). Learning Transferable Features with Deep Adaptation Networks.