Student Syllabus
OMsignal Block 2

The purpose of this document is to provide you with direction guidelines on how to take advantage of the unlabeled dataset for ECG signal processing in the context of the OMsignal project. This document contains a wide range of directions and one of your tasks will consist to either select one of those directions with a proper justification of your choice, or propose a new one.

I - Dealing with unlabeled data

There are two main axes for handling unlabeled data in general within the context of a machine learning project where some few labeled samples are available. The first one is based on an unsupervised pretraining setting while the second one relies on a semi-supervised setting. In what follows, we delve into the details of each axis.

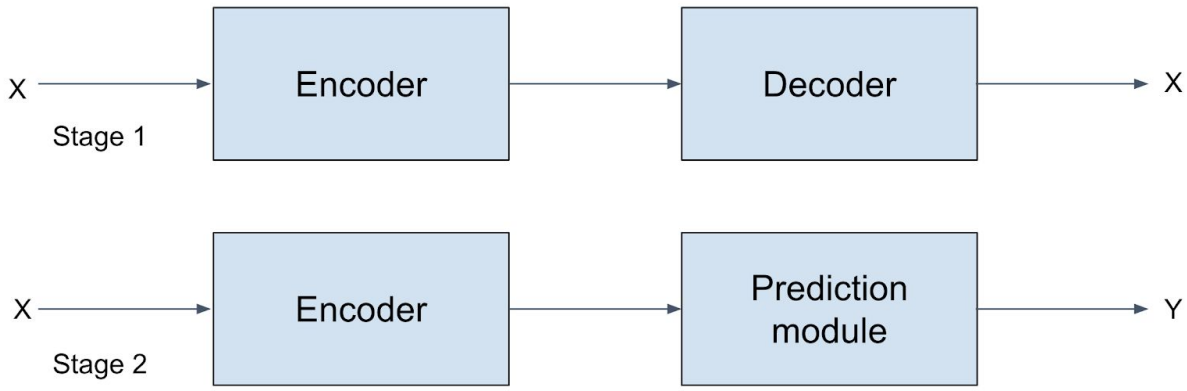
II - Unsupervised Pretraining

The approach here consists of two stages:

- First, you may consider training an Autoencoder model using the unlabeled data. During the training of the autoencoder, it may be interesting to consider a **denoising setting** where noise is added to the input signal and the autoencoder is forced to reconstruct the original signal. However, this setting supposes that you have at hand a **suitable function** for altering the input data.

- Next, use the encoder part of the autoencoder model as a feature extractor and rely on the extracted features for performing supervised tasks using labeled data. Here, you have two options for the feature extractor:
    - **Use it AS-IS**: when learning the supervised tasks, do no no alter the weights of the encoder (a.k.a. the feature extractor).
    - **Fine-tuning**: you can finetune the weight of the encoder. If so, it is important to keep in mind that the learning rate related to the parameters of the encoder **should be much smaller than** the one of the parameters involved in the

supervised setting in order to not completely erase the information gained from the unsupervised training.



III - Semi-supervised approach

Unlike the previous approach, the semi-supervised approach consists in only one stage:

- Use the unlabeled and the labeled data **jointly** in order to train a global architecture.

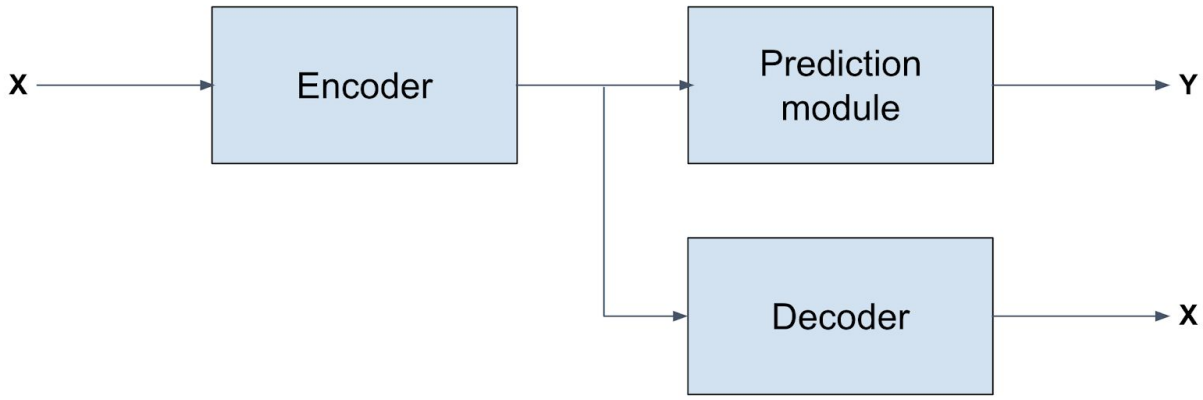
In this setting, it is important to properly manage the situation in which the underlying dataset is **unbalanced** (i.e. the amount of unlabeled data is significantly larger than the amount of labeled data). For example, special care is required when **constructing the mini-batches** for training. There are at least two options that can be considered:

- Alternate the training of the unsupervised task and the supervised ones, by alternating the mini-batches (with each mini-batch containing either only unlabeled data or only labeled data). Given that the unsupervised task has much more data than the supervised ones, limit the number of mini-batches for the unsupervised task to match that of the supervised ones.
- Similar to the previous bullet point, but this time considering all the mini-batches of the unsupervised task but with a weight downsampling the contribution of the unlabeled examples to the parts of the architecture shared by the unsupervised and supervised tasks.

There are several ways of performing semi-supervised learning in the literature. In this document, we only focus on some of them:

- ## Class 1

  This class contains approaches where a specific branch is added in the network architecture for handling unlabeled data. For example, we can consider an architecture where a decoder is added to reconstruct the data from the latent representations used for handling the supervised tasks. Architectures like Ladder Networks fall into this category.

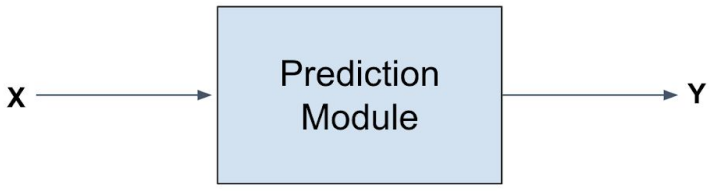


  Note that, **only labeled samples are used** in the loss involving Y.

  Reference: Ladder Networks: https://arxiv.org/pdf/1507.02672.pdf

- ## Class 2

  This class contains approaches which solely involve adding an additional loss term to the training of a neural network, and otherwise leaving the training and model unchanged from what would be used in the fully-supervised setting.



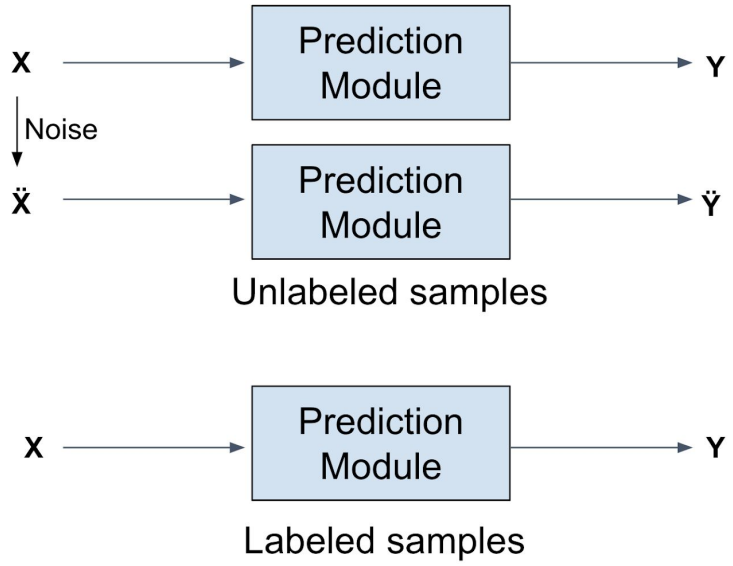
  Some hypotheses are made regarding the outputs of unlabeled data. These hypotheses lead to additional loss terms that are added to the loss of the supervised tasks as **regularized terms**.

This class can furthermore be subdivided into several subgroups according to the nature of the hypothesis made:

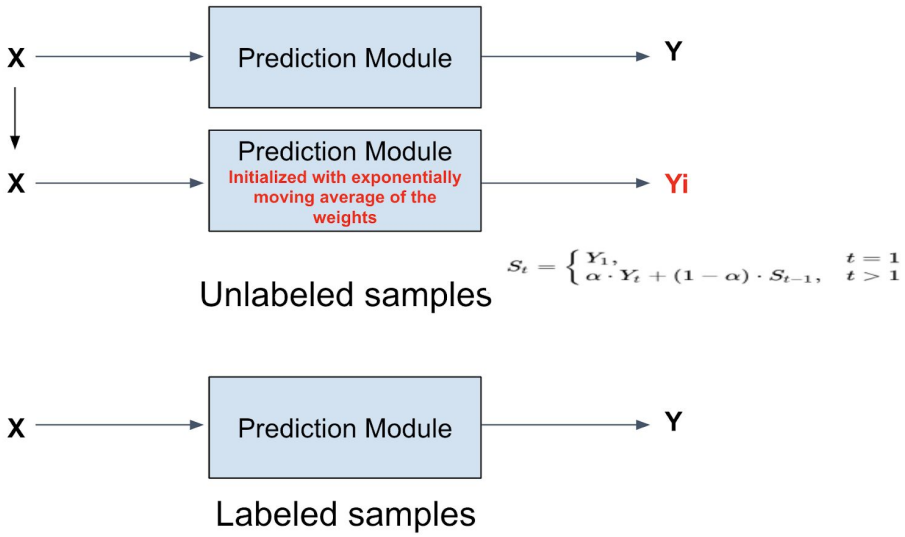
- **Consistency regularization:** describes a class of methods with the following intuitive goal: realistic and small perturbations $x \rightarrow \hat{x}$ of unlabeled data points x should not significantly change the output of $f_\theta(x)$ of the network (where θ represents the weights of the network). Generally, this involves minimizing $d(f_\theta(x), f_\theta(\hat{x}))$ where $d(\cdot, \cdot)$ measures a distance between the network's outputs, e.g. **mean squared error (for regression) or Kullback-Leibler divergence (for classification tasks)**. This supposes that you are able to perform **realistic perturbations** of the input data without affecting the associated labels. Typically the gradient through this consistency term **is only back propagated through $f_\theta(\hat{x})$**.
  - Note that this approach can be applied to both the labeled and unlabeled data.

  Several approaches have been designed in this subgroup such as:

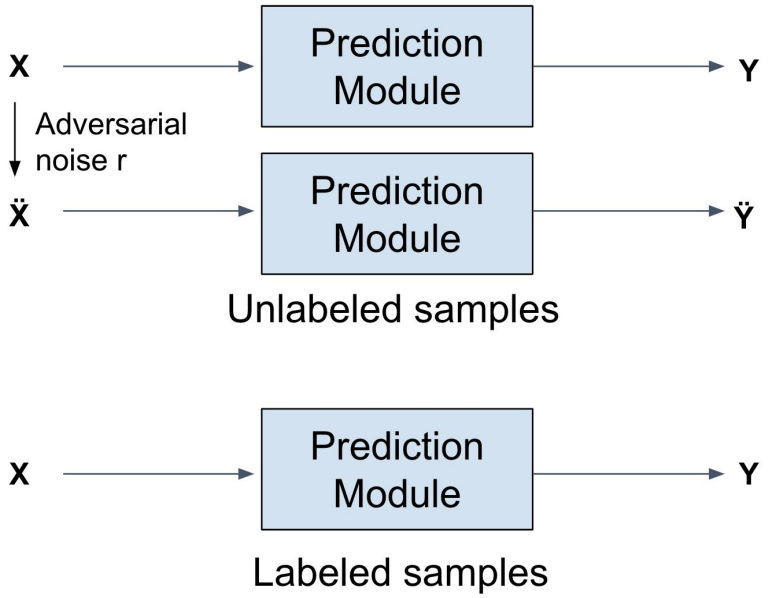
  - Stochastic perturbations: the straightforward application of consistency regularization is thus minimizing $d(f_\theta(x), f_\theta(\hat{x}))$ for unlabeled data x where in this case $d(\cdot, \cdot)$ is chosen to be the mean squared error. This distance term is added to the classification loss as a regularizer, scaled by a weighting hyperparameter.



- Mean Teacher: a difficulty with the previous approach is that it assumes that a perturbed (through noise addition) version of an input example will have an identical (or very close) output as the original input. This hypothesis is not easy to verify. "Mean Teacher" instead proposes to keep the input fixed, and uses a prediction function parametrized by θ', an exponentially accumulated average of θ over training to generate a perturbed output. As previously, the mean squared error d(fθ(x), fθ'(x)) is added as a regularization term with a weighting hyperparameter.



- Virtual Adversarial Training: instead of relying on the built-in stochasticity of fθ(x), Virtual Adversarial Training (VAT) directly approximates a tiny perturbation r to add to x which would most significantly affect the output of the prediction function. Consistency regularization is then applied to minimize d(fθ(x), fθ(x + r)) with respect to θ, effectively using the "clean" output as a target given an adversarially perturbed input. VAT is inspired by adversarial examples, which are natural datapoints x which have a virtually imperceptible perturbation added to them which causes a trained model to misclassify the datapoint.



- **Entropy Based approaches:** A simple loss term can be applied to unlabeled data to encourage the network to make "confident" (low-entropy) predictions for all examples, regardless of the actual class predicted. Assuming a categorical output space with K possible classes, this gives rise to the "entropy minimization" term: **$- \sum_{1:k} f_\theta(x)_k \log f_\theta(x)_k$** which is added to the supervised loss. On its own, entropy minimization has not been shown to produce competitive results. However, combined with VAT, this can gives rise to state-of-the-art results.

- **Pseudo Labels:** Pseudo-labeling is a simple heuristic which is widely used in practice, likely because of its simplicity and generality – all that it requires is that the model provides a probability value for each of the possible labels. It proceeds by producing "pseudo-labels" for the unlabeled data using the prediction function itself over the course of training. Pseudo-labels which have a corresponding **class probability which is larger than a predefined threshold** are used as targets for a **standard supervised loss function applied to unlabeled data**. While intuitive, it can nevertheless produce incorrect results when the prediction function produces unhelpful targets for the unlabeled data. Pseudo-labeling is quite similar to entropy regularization, in the sense that it encourages the model to produce higher-confidence (lower-entropy) predictions for unlabeled data. However, it differs in that it only enforces this for data points which already had a low-entropy prediction due to the confidence thresholding.

References

Stochastic Perturbations: https://arxiv.org/pdf/1606.04586.pdf
Mean Teacher: https://arxiv.org/pdf/1703.01780.pdf
Virtual Adversarial Training: https://arxiv.org/pdf/1704.03976.pdf
Pseudo Labeling: http://deeplearning.net/wp-content/uploads/2013/03/-pseudo_label_final.pdf