TEMPORAL ENSEMBLING FOR SEMI-SUPERVISED LEARNING

Introduction

Self-Ensembling

Self Ensembling is a technique that achieved state of the art results in the area of semi-supervised learning. It comes from a relatively simple idea, that we can use an ensemble of the previous outputs of a neural network as an unsupervised target. The two different methods are Π-Model and temporal ensembling.

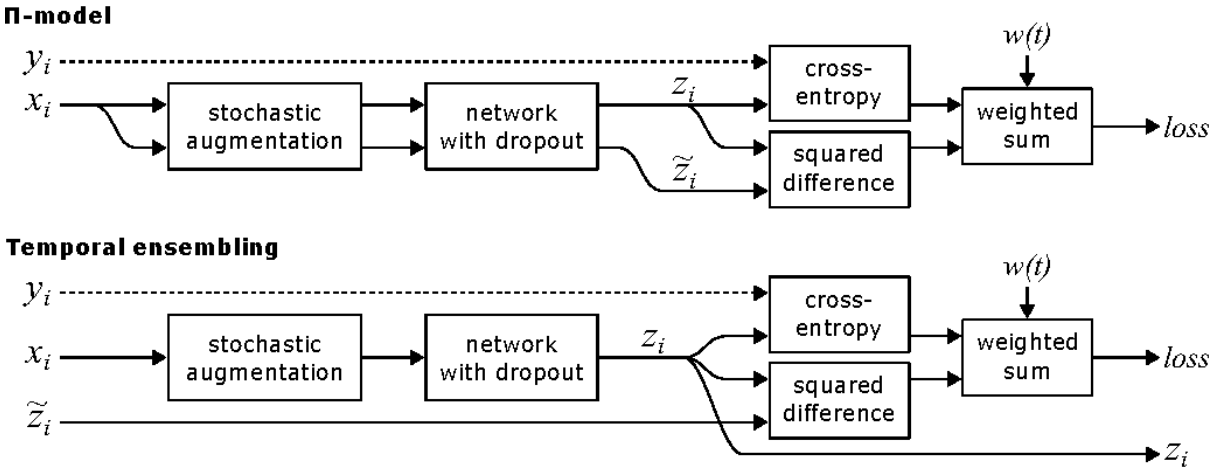


Fig: **PI Model and Temporal Ensembling**

Pi Model

In the Pi Model the training inputs x_i are evaluated twice, under different conditions of dropout regularization and augmentations, resulting in tow outputs z_i and $\tilde{z}_i$.

The loss function here includes two components:

- **Supervised Component**: standard cross-entropy between ground-truth y_i and predicted values z_i (only applied to labelled inputs).
- **Unsupervised Component**: penalization of different outputs for the same input under different augmentations and dropout conditions by mean square difference minimization between z_i and $\tilde{z}_i$. This component is applied to all inputs (labelled and unlabelled).

Advantages:

- This two components are combined by summing both components and scaling the unsupervised one using a time-dependent weighting function $w(t)$.

- The augmentations and dropouts are always randomly different for each input resulting in different output predictions. Additionally to the augmentations, the inputs are also combined with random Gaussian noise to increase the variability.

- The weighting function $w(t)$ will be described later since is also used by temporal ensembling, but it ramps up from zero and increases the contribution from the unsupervised component reaching its maximum by 80 epochs. This means that initially the loss and the gradients are mainly dominated by the supervised component (the authors found that this slow ramp-up is important to keep the gradients stable).

Disadvantages:

- The only problem occurs is that we faced during Π-Model is that it relies on the output predictions that can be quite unstable during the training process. To combat this instability the authors propose the temporal ensembling.

Temporal Ensembling

Temporal Ensembling was introduced by Timo Aila and Samuel Laine (both from NVIDIA) in 2016. It generally means that they compare the network current outputs (post-softmax) to a weighted sum of all its previous outputs. Previous outputs are gathered during training: in an epoch, each input is seen once and its corresponding output is memorized to serve as comparison later.

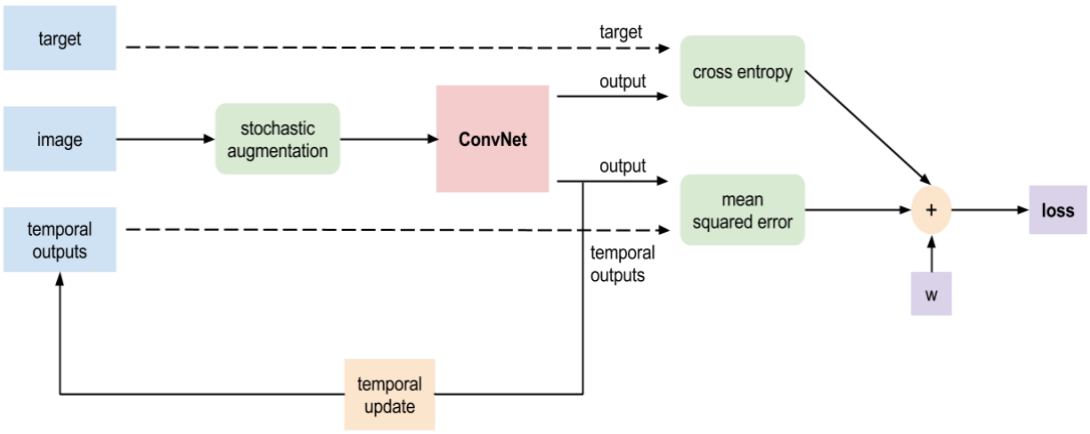


Fig: Temporal Ensembling Mechanism

Step 1: To resolve the problem of the noisy predictions during train, first we aggregate the predictions of past predictions into an ensemble prediction.

Step 2: The predictions z_i are forced to be close the ensemble prediction $\tilde{Z}_i$ that is based on previous predictions of the network. This algorithm stores each prediction vector z_i and in the end of each epoch, these are accumulated in the ensemble vector $\tilde{z}_i$ by using the formula: where α is a term that controls how far past predictions influence the temporal ensemble.

Step 3: This vector contains a weighted average of previous predictions for all instances, with recent the ones. As comparing to both the ensemble training targets, they need to be scaled by diving them by $(1-\alpha^t)$

Step4: In this, Z and $\tilde{z}_i$ are zero on the first epoch, since no past predictions exist.

The advantages of this algorithm when compared to the PI Model are:

- Training is approximately 2x faster (only one evaluation per input each epoch)
- The $\tilde{z}_i$ is less noisy than in PI Model.

The disadvantages of this algorithm when compared to the PI Model are:

- The predictions need to be stored across epochs.
- A new hyper parameter α is introduced.

Semi-Supervised Learning

Semi-Supervised Learning algorithms try improving traditional supervised learning ones by using unlabelled samples. This is very interesting because in real-world there are a big amount of problems where we have a lot of data that is unlabelled. There is no valid reason why this data cannot be used to learn the general structure of the dataset by supporting the learning process of the supervised training.

Code:

```
#  Editable variables

num_labeled_samples = 1000

num_validation_samples = 400

batch_size = 50

epochs = 300

max_learning_rate = 0.001

initial_beta1 = 0.9

final_beta1 = 0.5

checkpoint_directory = './checkpoints/PiModel'

tensorboard_logs_directory = './logs/PiModel'
```

temporal(1).py - C:\Users\ADMIN\AppData\Loc

File Edit Format Run Options Window Help

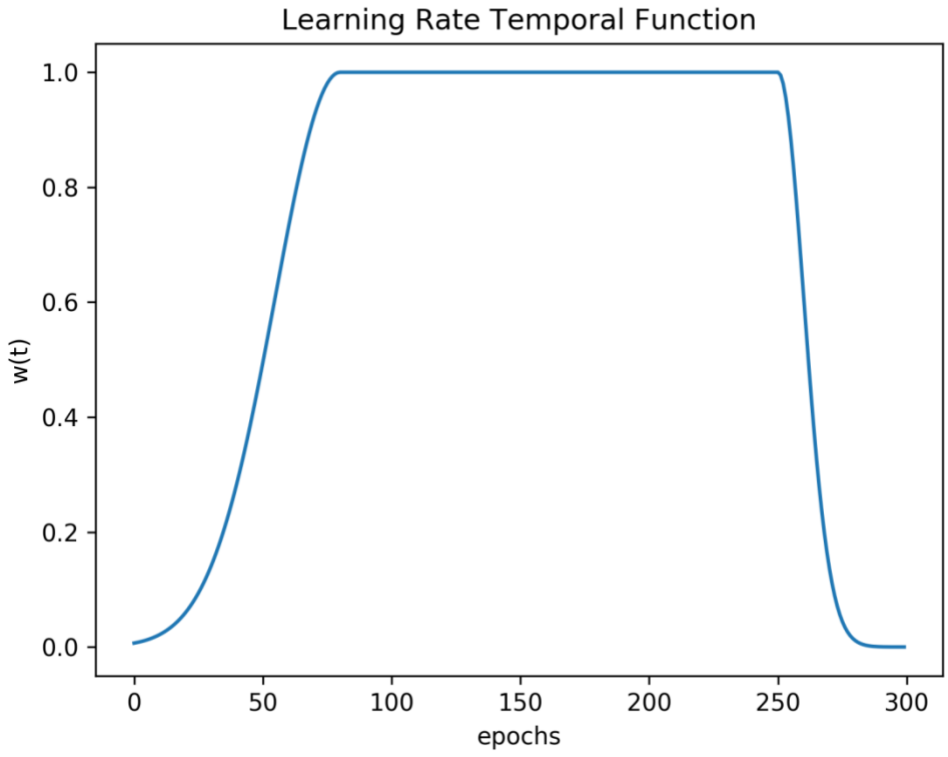
```
#  Editable variables
num_labeled_samples = 1000
num_validation_samples = 200
batch_size = 50
epochs = 300
max_learning_rate = 0.001
initial_beta1 = 0.9
final_beta1 = 0.5
checkpoint_directory = './checkpoints/PiModel'
tensorboard_logs_directory = './logs/PiModel'
```

IDLE Shell 3.9.2

File Edit Shell Debug Options Window Help

```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64 bi
t (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\ADMIN\AppData\Local\Programs\Python\Python39\temporal(1
).py
>>>
```

Ln: 5 Col: 4



Train temporal ensembling model

```
# Editable variables

num_labeled_samples = 2000

num_validation_samples = 500

num_train_unlabeled_samples = str(input("NUM_TRAIN_SAMPLES -
num_labeled_samples - num_validation_samples"))

batch_size = 100

epochs = 300

max_learning_rate = 0.0002 # 0.001 as recomended in the paper leads to unstable training.

initial_beta1 = 0.9

final_beta1 = 0.5

alpha = 0.6

max_unsupervised_weight = 30 * num_labeled_samples /
str(input("NUM_TRAIN_SAMPLES - num_validation_samples"))

checkpoint_directory = './checkpoints/TemporalEnsemblingModel'

tensorboard_logs_directory = './logs/TemporalEnsemblingModel'

print(" NUM_TRAIN_SAMPLES - num_validation_samples : ")
```

traintemporal(2).py - C:\Users\ADMIN\AppData\Local\Programs\Python\Python39\Lib

File Edit Format Run Options Window Help

```
 # Editable variables
num_labeled_samples = 2000
num_validation_samples = 1000
num_train_unlabeled_samples = str(input("NUM_TRAIN_SAMPLES - num_labeled_samples - num_validation_samples"))
batch_size = 100
epochs = 300
max_learning_rate = 0.0002 # 0.001 as recomended in the paper leads to unstable training.
initial_beta1 = 0.9
final_beta1 = 0.5
alpha = 0.6
max_unsupervised_weight = 30 * num_labeled_samples / str(input("NUM_TRAIN_SAMPLES - num_validation_samples"))
checkpoint_directory = './checkpoints/TemporalEnsemblingModel'
tensorboard_logs_directory = './logs/TemporalEnsemblingModel'
print(" NUM_TRAIN_SAMPLES - num_validation_samples : ")
```

IDLE Shell 3.9.2

File Edit Shell Debug Options Window Help

```
Python 3.9.2 (tags/v3.9.2:1a79785, Feb 19 2021, 13:44:55) [MSC v.1928 64
bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
= RESTART: C:\Users\ADMIN\AppData\Local\Programs\Python\Python39\Lib\trai
ntemporal(2).py
NUM_TRAIN_SAMPLES - num_labeled_samples - num_validation_samples
NUM_TRAIN_SAMPLES - num_validation_samples
```

Ln: 6 Col: 42