# TEMPORAL ENSEMBLING FOR MULTI-INSTANCE LEARNING

**Luciano de la Iglesia & Wilson Tang**
Department of Computer Science
University of Washington
Seattle, WA 98105, USA
{lucianod,wtang06}@cs.washington.edu

## ABSTRACT

Temporal ensembling is a successful self-ensembling technique for leveraging unlabeled data for supervised training. It was introduced in the domain of image classification, but we expand it to binary multi-instance learning on images. We evaluate results on two datasets, MNIST-bags and breast cancer samples, and show that our model performs similarly or worse than the model on which it is based. Although we intuitively believe that temporal ensembling should be applicable to more domains, we were not able to make it work for multi-instance learning.

## 1 INTRODUCTION

Deep learning has revolutionized artificial intelligence, but its reliance on large datasets is problematic given the high cost and lack of reliability of having humans label data. One solution is semi-supervised learning, a burgeoning field which seeks to solve traditional supervised problems such as image classification with datasets that are only partially labeled.

A recent strategy is temporal ensembling, which drives models to produce more similar outputs for the same inputs over time, regardless of whether the inputs have a corresponding label. Laine & Aila (2017) introduce this technique and apply it to image classification. In this paper, we extend it to the problem of multi-instance learning (MIL) (Maximilian Ilse, 2018). Both these papers were covered in class. Unfortunately, in our experiments, our model does not outperform traditional MIL models.

### 1.1 TEMPORAL ENSEMBLING

Ensembling is a popular machine learning technique in which multiple models are combined (ensembled) to yield better results than any single model. Temporal ensembling is a self-ensembling technique which combines a single model's predictions over time as it is trained to leverage unlabeled data for supervised tasks (Laine & Aila, 2017).

Figure 1 shows the temporal ensembling training regime. We start with a dataset of points $x_i$, some of which have labels $y_i$. The task in Laine & Aila (2017) is image classification. For each training batch $B$, each point $x_i \in B$ passes through a stochastic input augmentation function followed by a convolutional neural network, producing $z_i$, the vector containing predicted class probabilities for that point. For points $x_i$ with a label $y_i$, a standard cross-entropy loss term is used. This is added to the following loss term for all points $x_i$, unlabeled and labeled:

$$w(t)\frac{1}{C|B|}\sum_{i\in B}\|z_i - \tilde{z}_i\|^2$$

$\tilde{z}_i$ is the temporal ensemble, an exponential moving average of vectors $z_i$ at previous training steps. $w(t)$ is a weight ramp function that increases over time to decrease the impact of the untrained model's predictions at the beginning of training. $C$ is the number of classes.

Intuitively, this drives the model to produce more similar results for the same points over time, providing invariance to the stochastic input augmentation and dropout in the network, even for un-

labeled points. We refer readers to Laine & Aila (2017) for a more detailed description of temporal ensembling.

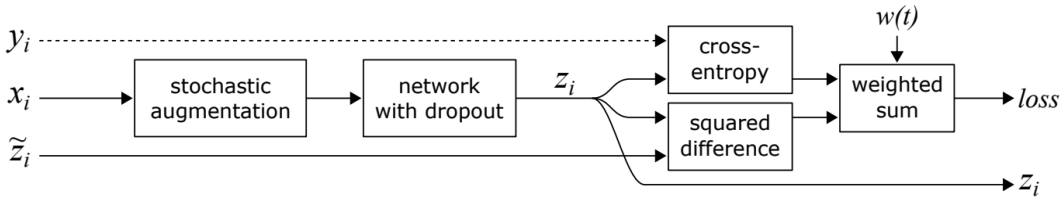



Figure 1: The temporal ensembling training pipeline, from Laine & Aila (2017).

### 1.2 Multi-Instance Learning

Multi-Instance Learning (MIL) is a class of supervised learning problems in which a single label is predicted for a bag of data points. This is particularly useful in the medical diagnoses domain because images are typically very high resolution, making running the full image through a Convolutional Neural Network (CNN) computationally infeasible. Instead, the image is cropped into sections and the individual crops are fed through a CNN. In short, a medical diagnosis image is cropped into sections and a single label (positive or negative) is given. In the context of breast cancer, there would be a benign or malignant label associated with each bag of images for an individual.

The approach to a general MIL problem is to assign a label for each bag of points. For the binary classification example, we can model the problem as 0 if all the labels in the bag are 0 and 1 otherwise (Maximilian Ilse (2018)):

$$Y = \begin{cases} 0 & \text{iff } \sum_k y_k = 0 \\ 1 & \text{otherwise} \end{cases}$$

There are two popular approaches to MIL: instance-level and embedding-level (Maximilian Ilse, 2018). In the instance-level approach, an instance-level classifier is applied to each sample in the bag and then aggregated by an MIL pooling function to obtain a score for the whole bag. In the embedding-level approach, an MIL pooling function is applied to obtain a representation for the whole bag, then fed through an embedding-level classifier to obtain a score for the bag. Maximilian Ilse (2018) combines these two approaches and uses attention as the MIL pooling function to obtain better results than instance-level or embedding-level would achieve on their own. Mathematically, they have:

$$\left| S(X) - g\left( \max_{\mathbf{x} \in X} f(\mathbf{x}) \right) \right| < \varepsilon,$$

where $f$ is the embedding function, parametrized by a neural network, $max$ is the parallel to the attention model MIL pooling function, and $g$ is another neural network. $S(X)$ is the true score to compare the output to. We refer readers to Maximilian Ilse (2018) for a more detailed description of attention-based deep MIL.

### 1.3 Motivation

Medical images are costly to label. Strong labels would label where exactly in a medical diagnosis image is causing the diagnosis. But attention-based deep MIL solves this problem by only needing one label for the whole image and uses attention to learn where the key instances in the image are. We take this work one step further and attempt to learn on datasets where only some of the bags are labeled. We combine temporal ensembling with attention-based deep MIL.

## 2 Methods

Our starting point is the open-source implementation of Maximilian Ilse (2018) from the authors on Github (`github.com/AMLab-Amsterdam/AttentionDeepMIL`). It is implemented in the PyTorch framework. Our code is at `github.com/Lucianod28/Temporal-MIL`. Our

primary task was to update the loss function in the model to have a temporal ensembling loss. In the loss function, we added a temporal loss component that compares the current prediction to the current ensemble prediction. To keep track of the current ensemble predictions, we added two new variables, $Z$ and $\tilde{z}$ to keep track of the moving average of the ensemble predictions. We also updated the train function to keep track of 4 new hyperparameters: $percentage\_labeled$, $max\_unsupervised\_weight$, $epoch\_with\_max\_rampup$, and $\alpha$.

$percentage\_labeled$ is the percentage of the dataset for which labels were available during training. $max\_unsupervised\_weight$ controls the weight of the temporal ensembling loss term. The value used by Laine & Aila (2017) did not work for our domain, so we experimented with different values for this hyperparameter. $epoch\_with\_max\_rampup$ controls at which epoch the unsupervised weight ramping function, $w(t)$, reaches its maximum. This is necessary to prevent the ensemble prediction to have too much weight at the beginning of training the model. $\alpha$ was the weight used to calculate the exponential moving average of predictions for each point, $Z$.

### 2.1 MNIST-BAGS

We ran baselines on the MNIST-bags dataset. In the MNIST-bags problem, each bag contains samples where each sample is a handwritten digit. A bag is tagged as 1 if the bag contains one or more 9's and 0 otherwise. We train our temporal ensembling-based model on 500 train bags and 50 test bags. To gather baselines, we run the fully supervised MIL model over 100%, 50%, 30%, 20%, 10%, and 5% out of 500 train bags and gather error rates on the 50 test bags. We ran the temporal ensembling-based model with 50% of the training bags labeled. So, the model was trained on 250 labeled bags and 250 unlabeled bags. To find the best error rate we could, we search over the hyperparameters of $max\_unsupervised\_weight$ and $epoch\_with\_max\_rampup$. We repeat the process for 30%, 20%, 10%, and 5%. We finally compare our test error rates on 50 bags between the fully supervised model and temporal-ensembling-based model.

### 2.2 ROBUSTNESS TO MISLABELED DATA

We replicated the Laine & Aila (2017) robustness to mislabeled data experiment. We used the MNIST-bags dataset and chose some fraction of the training data to randomly relabel, according to the $Bernoulli(0.5)$ distribution. We experimented with 90%, 80%, 50%, and 20% randomly labeled data, with the rest of the training points retaining their original label.

### 2.3 BREAST CANCER

The open-source implementation of Maximilian Ilse (2018) only contains code for the MNIST-bags problem. To test our model on the breast cancer dataset, we implemented our own DataLoader for the breast cancer dataset. We obtained the breast cancer dataset from `bioimage.ucsb.edu/research/bio-segmentation`. This dataset consists of 58 896x768 images. Each image is tagged benign or malignant depending on if they contained breast cancer cells or not. We crop each image into 32x32 patches, resulting in 672 patches per bag. We discarded images containing 75% or more white pixels. We then put these bags of images into a PyTorch DataLoader to feed in to our model. Our test set contained 20% of the dataset. We also implemented the suggested model from Maximilian Ilse (2018) for histopathology datasets. This model was too large for our 8GB GPU, so we decreased the number of convolutional channels. The model still took a long time to train, so we decided to only run the temporal ensembling-based model on 50% labeled images. Since the dataset has only 58 bags, we thought splitting into smaller labeled percentages while maintaining class balance and a test set would make our data splits too small.

## 3 RESULTS

### 3.1 MNIST-BAGS

Figure 2 shows results for the MNIST-bags dataset. Temporal ensembling has significantly higher test error than the baseline model at 30% and and 20% labeled, with similar performance on 5%, 10%, and 50%. Figure 3 shows the effect on training loss and test error of varying the

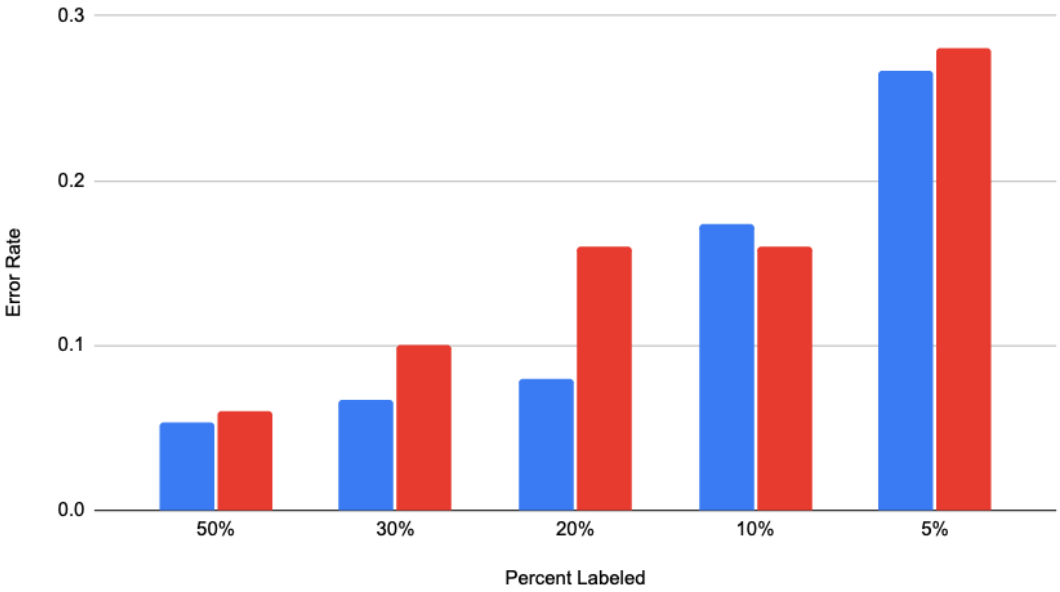
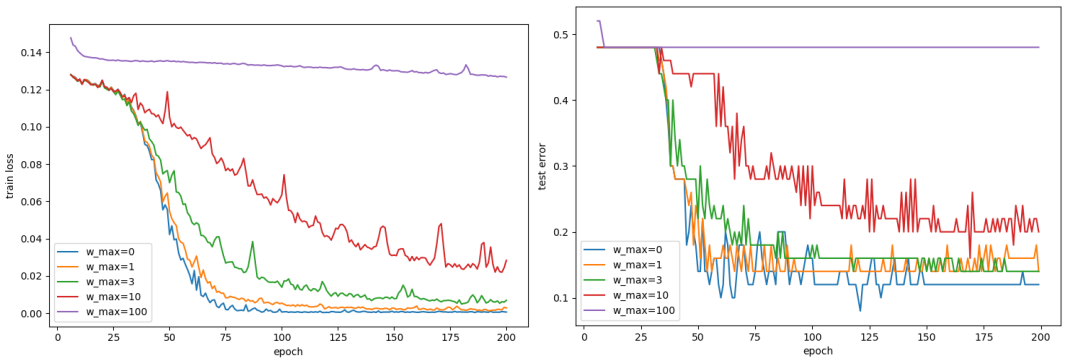



Figure 2: Error rate on the test set for the baseline model (blue) and the temporal ensembling model (red) on the MNIST-bags dataset. Results are shown for 50%, 30%, 20%, 10%, and 5% of the dataset being labeled.

$max\_unsupervised\_weight$ hyperparameter. This hyperparameter greatly influences the performance of the model as it determines the relative importance of the temporal ensembling loss term.




(a) Train loss vs. epoch on different max weights

(b) Test error vs. epoch on different max weights

Figure 3: Train loss and test error on 20% labeled data with different values for max unsupervised weights

Figure 3 shows the train loss and test errors on 20% labeled training data for various max unsupervised weights. This shows how the temporal ensembling model performs under different hyperparameters. We show that even under different hyperparameters for the temporal ensembling model, the train loss and test error don't converge to a rate any better than without temporal ensembling. A max weight of 0 means that there was no temporal ensembling loss and was fully supervised over 20% of the data.

Figure 4 shows the robustness of these models to mislabeled data on the MNIST-bags dataset. Our temporal ensembling model outperforms the baseline when 90% of the data is randomly labeled, but performs as well as or worse than the baseline model on 80%, 50%, 20%, and 10%.

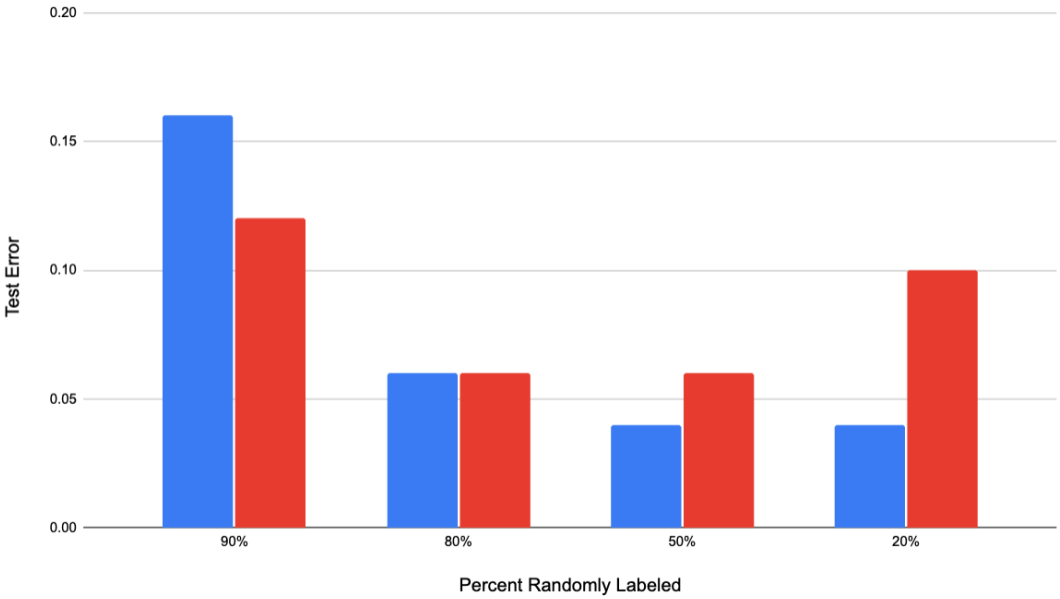



Figure 4: Error rate on the test set for the baseline model (blue) and the temporal ensembling model (red) on the MNIST-bags dataset. The x-axis is the percent of the training data that was randomly labeled.

### 3.2 Breast Cancer

For the breast cancer dataset at 50% labeled, the baseline model and temporal ensembling models both achieved test loss of 5.6%.

## 4 Discussion

Despite Laine & Aila (2017) only showing results on image classification, temporal ensembling is not inherently domain-specific, so we expected it to yield similar results on MIL as on image classification, particularly given the similarity of these domains. However, we found temporal ensembling was not effective for the datasets we experimented with. For MNIST-bags, we found our error rates slightly increased for all percentage labeled except for the 10%, but the 10% only improved by a slight 1.33%. We also experimented with different hyperparameters, primarily the max unsupervised weight. We found that no value for the max weight would make our model converge to a better loss. We conclude that temporal ensembling was not effective for MNIST-bags. We then tested the model on the breast cancer dataset and found the same result. Based on Figure 3, it seems our model is not learning from the unlabeled data. If anything, we are making our losses worse by adding more weights to our unsupervised loss. But setting the weights too low would not add any insight from the unlabeled data.

When we discussed Laine & Aila (2017) in class, some students mentioned that they would have liked to see results on multiple domains. This was part of the inspiration for this project, as we thought it should be relatively simple to generalize this to other domains. However, perhaps Laine & Aila (2017) chose to only show the tasks where their algorithm was most effective.

While temporal ensembling did not work for us, it is possible that we did not find the right hyperparameters or that there are other domain-specific adaptations that need to be made. It is hard to know all the optimizations that were made in the Laine & Aila (2017) and Maximilian Ilse (2018) papers and how these may affect our results. We remain optimistic that under the right conditions, temporal ensembling can be extended to other domains.

## REFERENCES

Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. 2017.

Max Welling Maximilian Ilse, Jakub M. Tomczak. Attention-based deep multi-instance learning. 2018.