Investigating CoTTA: Validating Real-Time Neural Network Adaptations

Ansh Mujral
amujral@ucsd.edu

Ifunanya Okoroma
iokoroma@ucsd.edu

Keenan Serrao
kserrao@ucsd.edu

Nicholas Swetlin
nswetlin@ucsd.edu

Jun-Kun Wang
jkw005@ucsd.edu

Abstract

Machine learning models encounter the problem of distribution shifts, where the data used to train the model differs from the data used when the model is deployed at test time. When training new systems without access to the test data, it's important that these models still perform well under those circumstances. One approach to address this is Test-Time Adaptation with Pseudolabels (TTAPL). While this is a well-established solution, TTAPL needs better variations to produce more accurate and generalizable models. In this work, we investigate the model structures and consistency losses of CoTTA (Continual Test-Time Adaptation), one type of TTAPL. We develop three loss functions and test CoTTA on seven different models. From these experiments, we find that the loss functions we developed make better predictions compared to the original CoTTA and the model size used affects the efficacy of CoTTA.

Website: https://keenans04.github.io/cotta-pages/
Code: https://github.com/KeenanS04/cotta-variations/tree/main

1 Introduction

1.1 Test-Time Adaptation (TTA)

As machine learning becomes more widely used in today's world, there are various issues that arise with it. One of them being addressing of distribution shifts within data. Distribution shifts occur when data the model was trained on vastly differs from the data performed on at test time. These shifts can lead to miscalculations in models, making them perform worse and reducing their efficiency.

Self-driving cars is one example of the importance of addressing distribution shifts. They need to be able to adapt in real time to different lighting conditions. Otherwise, common environments with lots of rapid lighting changes, such as driving through a short tunnel or on a curving road during sunset, would be significantly more unsafe than stable lighting environments to have self-driving cars navigate. Besides self-driving cars, rapid adaptation is useful in many technologies: personalized medical imaging of tumors, analyzing market trends of specific stocks, and conversion of handwriting to digital text (OCR).

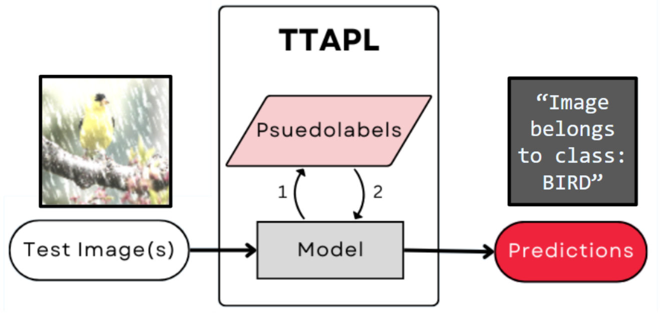
1.2 Test-Time Adaptation with Pseudo-Labeling (TTAPL)

When assessing distribution shifts in live settings, test data may additionally be *unlabeled* – i.e. the label is not known for any particular test data. Distribution-shifted, unlabeled test data can be handled by a class of TTA methods that use *pseudo-labeling*, or the creation of approximate test data labels to further train the model; we will refer to this class of methods as TTAPL.

Algorithm 1: General Form of Test-Time Adaptation with Psuedo-labels

```
Initialization: A source pre-trained model $f_{\theta_0}(x)$.
Input: Data $x$ from dataset $X$.
    1. Generate pseudo-label $\hat{y}$ for $x$.
    2. Update model weights $\theta$ from psuedo-labels $\hat{y}$ in some fashion.
Output: Updated model $f_\theta(x)$.
```

Figure 1: General TTAPL Algorithmic Structure



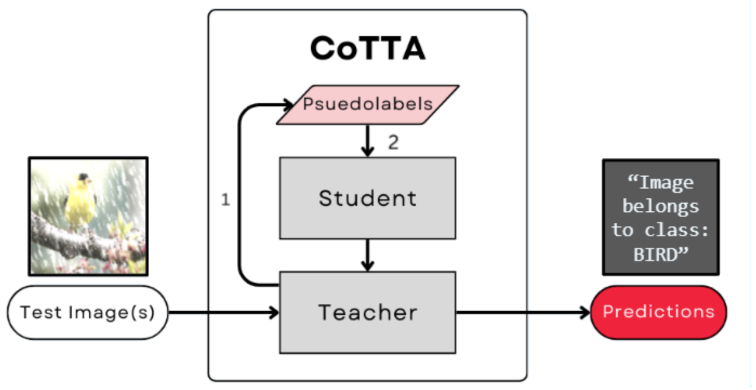
1.3 Continual Test-Time Adaptation (CoTTA)

For these adaptations with unlabeled test data, the concept of CoTTA, or "Continual Test-Time Adaptation" (Wang (2022)) proves to be useful, as it allows models to update their parameters in real time and adapt to new data on the fly.

Algorithm 2: Simplified Version of Continual Test-Time Adaptation

```
Initialization: A source pre-trained model $f_{\theta_0}(x)$, teacher model $f_{\theta'_0}(x)$ initialized from $f_{\theta_0}(x)$.
Input: For each time step $t$, current stream of data $x_t$.
    1. Augment $x_t$ and get weight and augmentation-averaged pseudo-labels from the teacher $f_{\theta'_t}$.
    2. Update student $f_{\theta_t}$ by consistency loss.
    3. Update teacher $f_{\theta'_t}$ by moving average.
Output: Prediction $f_{\theta'_t}(x_t)$; Updated student model $f_{\theta_{t+1}}(x)$; Updated teacher model $f_{\theta'_{t+1}}(x)$.
```

Figure 2: Simplified CoTTA Algorithmic Structure



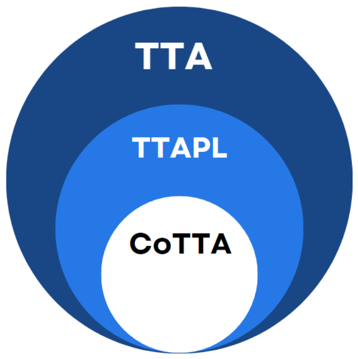
In essence, CoTTA operates by having two initally identical versions of the source model (dubbed the ”student” and ”teacher” models) train each other in real-time, immediately following each *data point*, not an entire dataset (as is common in other forms of TTAPL). The student model is more prone accepting to drastic changes, while the teacher model is quite ”skeptical”, integrating what the student tells it via a weighted average of the current teacher and student model weights. Together, this two-model system produces a clever teacher model, which will – in theory – supply more accurate predictions than the initial source model.

For the scope of our investigation, we will not dive too deep into the infrastructure of CoTTA. Our particular interest lies in the second step of the algorithm: updating the student model via ”consistency loss”. Our belief is that because this consistency loss affects how the student is updated and how it subsequently updates the teacher, performance is fairly situational; depending on the source model, CoTTA may achieve better performance with a different consistency loss than the default consistency loss explored in previous literature: the cross-entropy loss. Let $\hat{t}$ be the teacher's prediction vector and for a given data point x from dataset X, and let $\hat{s}$ be the student's prediction vector for that same data point; the cross-entropy version of a consistency loss would be structured as such:

$$\mathcal{L}_{\text{CE}}(\hat{t},\hat{s}) = \underbrace{-\sum_{c} \hat{t}_c \log \hat{s}_c}_{\text{Cross-Entropy Loss}}$$

With the consistency loss in mind, we aim to assess the robustness of CoTTA through the following questions: *Does the quality of CoTTA hold up on simple and complex models alike? How do different model types affect the performance of CoTTA? Are different consistency loss methods employed in CoTTA superior under different model types?*

Figure 3: Relationship between TTA, TTAPL, CoTTA



2 Methods

2.1 Experiment Structure

How we'll evaluate different models with various consistency loss functions will be with the standard machine learning image classification task of **CIFAR-10-to-CIFAR-10C** (Hendrycks (2019)). In this task, CIFAR-10 is a dataset consisting of 60,000 small 32×32 color images across 10 different classes. In contrast, CIFAR-10C is a variant of the CIFAR-10 test set where various algorithmically generated corruptions (such as noise, blur, weather effects, and digital distortions) have been applied. Algorithms perform well on this task if they are able to properly classify a large number of corrupted images from the CIFAR-10C test set.

For each combination of model and consistency loss function, we proceeded to run through the CIFAR-10-to-CIFAR-10C task, tracking the accuracy, precision, recall, and F1 score for each.

We expect several of our TTA variations to perform better (i.e. allow the models that TTA adapts to create higher quality predictions) than other forms of TTA, including original CoTTA, for reasons listed below in our methods section. We also expect to see that no single model type will be best for CoTTA's performance.

Note: All uses of CoTTA in this paper employ weight-averaging, augmentation-averaging, and stochastic restoration, as the inclusion of all of these methods found the greatest model improvements in Wang (2022). Additionally, to mirror methodology from previous literature, our CIFAR10-to-CIFAR10C benchmark only tests images of ”severity five”, instead of the whole image dataset.

2.2 Model Choice

We constructed Continual Test-Time Adaptation (CoTTA) with multiple loss functions and compared its performance with other Test-Time Adaptation (TTA) techniques in order to assess the resilience of CoTTA across diverse neural network designs. We aim to evaluate the relationship between model complexity and various loss functions in an adaptive learning environment.

RobustBench, a benchmark resource for assessing adversarial robustness, provided the neural network designs we employed. Our choice was influenced by availability and the requirement for different levels of complexity in the designs. In order to evaluate the ways in which various training paradigms interact with test-time adaptation, these included regular ResNet (R), WideResNet (WR), and PreActResNet (PR) architectures, with some models using other robustness training strategies, such as ”AugMix” and ”RLAT”. The architectures that were chosen were:

Table 1: Model Information; Models are ordered by ”size”, from fewest number of layers to greatest number of layers. Additionally, models are given aliases to use in later result tables.

Alias	**Model**	**Layers**	**Architecture**
M1	Kireev2021Effectiveness_Gauss50percent	18	PR
M2	Kireev 2021 Effectiveness_RLAT	18	PR
M3	Kireev2021Effectiveness_RLATAugMix	18	R
M4	Standard	28	WR
M5	Hendrycks2020AugMix_ResNeXt	29	R
M6	Addepalli2022Efficient_WRN_34_10	34	WR
M7	Hendrycks2020AugMix_WRN	40	WR

Each neural network architecture has its own way of structuring layers, which plays a role in how well it adapts during testing:

- ResNet (R) architectures use skip connections to maintain smooth gradient flow.
- WideResNets (WR) expand the number of channels, allowing for richer feature extraction.
- PreActResNets (PR) rearrange their layers by applying batch normalization and ReLU before convolutions, helping stabilize gradients.
- Models trained with RLAT focus on adversarial robustness.
- AugMix models, on the other hand, apply stochastic augmentation and consistency-based losses to boost generalization.

It is expected that these many structural and training differences affect how well each model integrates with CoTTA, making it possible to evaluate the impact of architectural complexity and robustness techniques on adaptation across multiple performance metrics. We anticipate that more robustness-trained models, such those that use RLAT or AugMix, would be more or less sensitive to Test-Time Adaptation, which could have an impact on how well they use CoTTA.

2.3 Loss Functions

In addition to the no adaptation, normal, and TENT variations of TTA to test from Wang (2022), we test CoTTA multiple times, each with a different consistency loss. In total, here are the eight forms of TTA we tested:

- No test-time adaptation.
- Vanilla test-time adaptation.
- TENT, fully test-time adaptation by entropy minimization.
- CoTTA, employing Cross-Entropy loss.
- COTTA, employing the Cosine Similarity.
- CoTTA, employing PolyLoss.
- COTTA, employing KL Divergence.
- COTTA, employing Self-Training Cross-Entropy loss.

We changed the consistency loss component of CoTTA to a variety of (four) different loss functions (Cosine, Polyloss, KL, Self-Training), selecting them according to their theoretical benefits and drawbacks in an adaptive learning environment.

Original: Cross-Entropy Loss
Since the cross-entropy loss is the typical classification loss function, it acts as the baseline. This function is very good at enforcing accurate predictions since it calculates the difference between the real labels and the projected probability distribution. We anticipate that this version would perform the best across all architectures because CoTTA was first created with cross-entropy loss in mind. Equation shown in Introduction.

PolyLoss
Additionally, we use PolyLoss, a cross-entropy extension that adds a second polynomial factor to lessen forecast overconfidence. The purpose of this function is to reduce overconfidence in ambiguous forecasts by establishing smoother decision boundaries. A loss function that steers clear of abrupt updates can be advantageous since TTA entails constant adaptation. For all testing with PolyLoss, we set the tunable hyperparameter $\epsilon = 0.1$.

$$\mathcal{L}_{\text{poly}}(\hat{t}, \hat{s}) = \mathcal{L}_{CE}(\hat{t}, \hat{s}) + \epsilon \cdot \hat{t}\,(1 - \text{softmax}(\hat{s}))\,,$$

Cosine Similarity
Another variant substitutes cosine similarity loss, a function frequently employed in embedding-based learning for cross-entropy loss. In certain cases, cosine similarity is advantageous since it calculates the angle between two vectors rather than their magnitude. This could be beneficial if the student and teacher prediction vectors happen to be close with respect to every entry in the vector (which is when the prediction vector and true label vector would be close in angle); this rewards "safer" student behavior with less extreme learning rather than more experimental student behavior. It could, however, find it difficult to successfully enforce accurate categorization in CoTTA since the loss function is not typically suited for probability distributions – which are not conceptualized as having "angles" between each other. Since some similarity preservation will still take place, we anticipate – in some cases – that it will perform worse than cross-entropy loss, but better than no TTA.

$$\mathcal{L}_{\cos}(\hat{t}, \hat{s}) = \frac{\hat{t} \cdot \hat{s}}{||\hat{t}||\ ||\hat{s}||}$$

KL Divergence
The difference between two probability distributions is then measured by the KL Divergence Loss, which we test next. Given that distribution adjustments inevitably result in modifications to class probabilities, KL Divergence might support incremental adaptation without producing significant prediction variations. However, if the model's initial predictions are already subpar, its dependence on actual probability distributions may be restrictive.

$$\mathcal{L}_{\text{KL}}(\hat{t}, \hat{s}) = \sum_{c} \hat{t} \log\left(\frac{\hat{t}}{\hat{s}}\right)$$

Self-Training Cross Entropy Loss We also test self-training cross-entropy loss, which makes use of pseudo-labeling by utilizing the student model's, and *exclusively* the student model's, own predictions as the basis for subsequent corrections. This loss function implementation is motivated by the work surrounding self-training loss functions (Wang (2022)), specifically the self-training cross-entropy loss, as, if the source model is trained using a cross-entropy loss, the optimality condition for this loss function is met and TTA utilizing a *self-training loss function* is plausible. One benefit of this approach is that it allows for unsupervised adaptation. However, there is also a chance that it will reinforce inaccurate predictions, particularly if early errors spread through Test-Time Adaptation.

$$\mathcal{L}_{\text{Self}}(\hat{s}) = \mathcal{L}_{\text{CE}}(\hat{s}, \hat{s})$$

3 Results

For each of these variations, we adapted a list of neural network classifiers and replicated the test-time adaptation benchmark CIFAR10-to-CIFAR10-C for multiclass classification. Below are four tables, each one for accuracy, precision, recall, and F1 score where:

- Each row represents a different neural network architecture.
- Each column represents a different TTA method.
- Each entry in the table describes an increase in prediction quality relative to no TTA, for a combination of architecture and TTA variation.

Table 2: Increase in Accuracy (%, Maximum Values Bolded)

	M1	M2	M3	M4	M5	M6	M7
None	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Normal TTA	5.08	9.45	2.83	23.08	4.72	**9.46**	3.78
TENT TTA	7.27	11.16	4.03	23.37	3.46	-13.52	4.46
CoTTA	8.37	**11.71**	5.20	26.53	6.44	-2.49	5.49
CoTTA-Poly	**8.99**	11.24	5.21	25.66	**6.62**	-7.44	**5.82**
CoTTA-KL	8.32	11.70	**5.22**	**26.58**	6.45	-2.58	5.51
CoTTA-Cosine	7.43	11.17	4.25	25.77	5.63	-5.65	4.62
Cotta-SelfTrain	-3.06	-17.67	-30.70	-1.74	-15.06	-45.56	-14.34

Table 3: Increase in Precision (%, Maximum Values Bolded)

	M1	M2	M3	M4	M5	M6	M7
None	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Normal TTA	2.43	5.50	1.14	8.60	1.72	**5.95**	0.83
TENT TTA	**6.23**	**8.83**	3.00	11.08	2.47	-3.52	2.63
CoTTA	5.62	7.60	3.28	11.92	3.35	-5.82	2.69
CoTTA-Poly	6.20	7.43	3.27	11.09	**3.50**	-11.70	**2.98**
CoTTA-KL	5.57	7.55	**3.32**	**12.00**	3.36	-5.95	2.70
CoTTA-Cosine	4.87	7.26	2.83	11.24	2.66	-9.01	2.03
Cotta-SelfTrain	-0.74	3.49	-0.73	-4.56	-0.11	-26.08	-1.15

Table 4: Increase in Recall (%, Maximum Values Bolded)

	M1	M2	M3	M4	M5	M6	M7
None	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Normal TTA	5.08	9.45	2.83	23.08	4.72	**9.46**	3.78
TENT TTA	7.27	11.16	4.03	23.37	3.46	-13.52	4.46
CoTTA	8.37	**11.71**	5.20	26.53	6.44	-2.49	5.49
CoTTA-Poly	**8.99**	11.24	5.21	25.66	**6.62**	-7.44	**5.82**
CoTTA-KL	8.32	11.70	**5.22**	**26.58**	6.45	-2.58	5.51
CoTTA-Cosine	7.43	11.17	4.25	25.77	5.63	-5.65	4.62
Cotta-SelfTrain	-3.06	-17.67	-30.70	-1.74	-15.06	-45.56	-14.34

Table 5: Increase in F1 Score (%, Maximum Values Bolded)

	M1	M2	M3	M4	M5	M6	M7
None	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Normal TTA	5.17	9.51	2.68	23.50	4.43	**9.88**	3.55
TENT TTA	7.46	11.29	3.96	24.06	3.20	-14.65	4.37
CoTTA	8.41	**11.71**	4.97	26.84	6.09	-2.29	5.28
CoTTA-Poly	**9.01**	11.33	4.97	25.97	**6.24**	-7.73	**5.59**
CoTTA-KL	8.36	11.68	**5.00**	**26.91**	6.10	-2.39	5.29
CoTTA-Cosine	7.53	11.25	4.18	26.14	5.32	-5.50	4.47
Cotta-SelfTrain	-3.05	-15.20	-29.08	-1.48	-12.24	-49.95	-12.74

Holding model architecture equal, we see that the improvement of accuracy, precision, recall, and F1 score for CoTTA variation predictions relative to source, with either the Cross-Entropy, KL, or PolyLoss form of consistency loss, provides similar quality (robustness within ± 1% for majority of models).

When the TTA loss is held equal, we see that model type has a clear effect (though non-directional) on the improvement of accuracy, precision, recall, and F1 score of CoTTA predictions relative to source. It is difficult to articulate a trend between size of model architecture and relative improvement of TTA method to source.

For all four evaluation metrics of accuracy, precision, recall, and F1 score – the same choices of TTA variation per model type prevail as the highest quality predictors.

4 Discussion

Across Consistency Losses

It is clear that a valid substitution to the cross-entropy consistency loss in the original implementation of CoTTA will yield similar, if not greater, TTA improvements as CoTTA with cross-entropy consistency loss. Of the ones tested here, **our methods of CoTTA with KL Divergence and CoTTA with PolyLoss show significant promise as valid substitutes for the consistency loss in CoTTA, dominating performance compared to previously established methods** (seen by the ratio of blue cells above vs. below the line partitioning the consistency losses into two groups).

Alternatively, self-training CoTTA is uniquely subpar in its prediction quality. We can only surmise that, in this case, a model meeting the optimality condition for a self-training loss function does not necessarily allow us to use such a condition within the CoTTA framework; the context for using a self-training method is more limiting than initially assumed for this experiment. Still, such a notion of a self-training loss function for a CoTTA framework is likely possible and practical given proper implementation.

Across Metrics

We used many common metrics related to classification quality: accuracy, precision, recall, F1 score. This implies that altering the consistency loss to a valid substitute of CoTTA does not tweak the relative weights of true positive, true negative, false positive, and false negative results, meaning that current biases or limitations of a model architecture will likely not significantly worsen or improve as a result of the introduction of a different consistency loss to a pre-existing CoTTA framework.

Across Architectures

There is no discernible directional trend for model architecture affecting the dominant TTA method. Certain model types just so happen to adapt better than others.

Different architectures pose a significant threat to the robustness of CoTTA as a method. For example, model M6 (Addepalli2022Efficient_WRN_34_10) had CoTTA-based TTA methods produce a negative effect on prediction quality across every metric, with normal TTA prevailing instead as the adaptation method with the highest quality predictions. Clearly this is not a result of using a WR architecture (or else M4 and M7 would have had similar trends), but rather M6 has other unique underlying architectural features that inhibit CoTTA performance. In essence, the lackluster performance of CoTTA-based methods on M6 is a clear example supporting that model choice remains the single greatest confounding factor for TTA.

Limitations

Limitations of the experimental approach include:

- A small sample size of select neural network architectures accessible on the RobustBench Github repository. Different choices of model architectures might influence the underlying improvement produced by TTA. These changes are supported and mentioned in pre-existing literature (Goyal (2022)).
- Metrics were evaluated on the CIFAR10-to-CIFAR10C benchmark non-continual task for TTA evaluation, which involves low resolution data sources. Other benchmarks for CoTTA methods, such as CIFAR100-to-CIFAR100C or ImageNet-to-ImageNetC, were not performed in the interest of expediting the experiment. Results may be different on higher resolution data, as corruptions or distribution shifts fundamentally change more cumulative data when applied to a higher resolution image.
- Not all possible hyperparameters were explored in the creation of this experiment. For example, the epsilon paramater for PolyLoss has a significant effect on the behavior of PolyLoss, so the TTA method of CoTTA with incorporated PolyLoss may need further examination of behavior with different values of epsilon.
- A non-gradual ordering of corruption types with only Severity 5 Corruptions were examined in this experiment. Lower severities of corruptions or the introduction of corruptions to a model gradually may influence the effectiveness of TTA methods involving CoTTA.

.

Future Work

Future work is needed to determine the efficacy of using CoTTA-based methods on architectures greater than 40 layers in size, and even beyond, to use within larger LLM structures that populate today's world.

In addition, more work on tweaking other parts of the CoTTA framework – such as the weighted averaging function that updates the teacher model – also has potential. This would get us one step closer to examining the *full* generalizability of CoTTA as a method.

5 Conclusion

CoTTA's generalizability is flexible across different dimensions. On one hand, the dimension of altering the anatomy of CoTTA itself (in our case, with the consistency loss) has demonstrated capability to produce higher quality methods of TTAPL under the CoTTA framework than existing methods – indicating flexibility within the CoTTA framework. On the other hand, CoTTA remains dramatically susceptible to the influence of model type in the quality of its adaptation, showing a lack of generalizability. Future work should be performed investigating the extent to which these dimensions of generalizibility extend to models of larger size, as to ascertain CoTTA's potential as a framework for LLMs.

6 Contributions

- Ansh Mujral: Implemented two different models from the integrated repository, analyzed result metrics, contributed to website.
- Ifunanya Okoroma: Implemented two different models from the integrated repository, contributed to paper and poster.
- Keenan Serrao: Restructured the original CoTTA repository to run on different architectures, contributed to website.
- Nicholas Swetlin: Implemented two different models from the integrated repository, contributed to paper and poster.

References

Goyal, Raghunathan Kolter, Sun. 2022. "Test-Time Adaptation via Conjugate Pseudo-labels." In *Neural Information Processing Systems (NeurIPS)*. [Link]

Hendrycks, Dietterich. 2019. "Benchmarking Neural Network Robustness to Common Corruptions and Pertubations." [Link]

Wang, Gool Dai, Fink. 2022. "Continual Test-Time Domain Adaptation." [Link]

Appendix

Adapted from our UCSD Capstone Quarter II Proposal

As machine learning becomes more widely used in today's world, there are various issues that arise with it, one of them being the presence of distribution shifts within data. Distribution shifts occur when data the model was trained on vastly differs from the data performed on at test time. These shifts can lead to miscalculations in models, making them perform worse and reducing their efficiency. For example, self-driving cars need to be able to adapt in real time to different lighting conditions. Otherwise, common environments with lots of rapid lighting changes, such as driving through a short tunnel or on a curving road during sunset, would be significantly more unsafe than stable lighting environments to have self-driving cars navigate.

Of course, rapid adaptation is useful in many technologies other than self-driving cars: personalized medical imaging of tumors, analyzing market trends of specific stocks, and conversion of handwriting to digital text (OCR) are all excellent examples. For these adaptations, the concept of continual test time adaptation (CoTTA) proves to be extremely useful, as it allows models to update their parameters in real time and adapt to new data (Wang (2022)). We aim to answer this question with our quarter two project: *How can we train neural networks to make these adaptations occur faster?*

One promising framework that allows neural networks underlying these machines to rapidly adapt to data is called Continual Test-Time Adaptation (CoTTA). CoTTA involves a complex system involving several interactions between two models: A "student" model (f_Θ) and a "teacher" model (f'_Θ). In default CoTTA, the student model at time t is trained via a cross-entropy loss, which is dependent on both the teacher and student models: $\ell_{\Theta_t} = -\sum_c y'_{tc} \log(\hat{y}_{tc})$; where $\hat{y}_t$ is the direct student prediction at time t and y'_t is the teacher prediction with the option of being augmentation-averaged at time t (depending on the specific version of CoTTA). But if we substitute the student's loss with a different loss function, such as a self-training loss function for cross-entropy dictated by conjugate pseudo-labels (mentioned in Goyal (2022)), or with a cosine similarity, what would happen? Would we see an increase or decrease in CoTTA's ability to adapt? Previous work on modifying CoTTA has not been substantially explored, and, combined with the fact that the specific loss functions we wish to investigate are either ubiquitously used in machine learning or satisfy other optimality conditions from machine learning adaptation frameworks, there is significant motivation to explore variations of CoTTA.

We propose to test the adaptive abilities of several “variations” of CoTTA, each of which is the same as the vanilla version of CoTTA with the replacement of the student’s loss function with a new loss function:

- Self-training loss function for cross-entropy loss.
- Cosine similarity between student and teacher predictions.
- A custom self-training loss function satisfying Goyal et al.’s expanded conjugate loss form.

For each variation, we will adapt a source model and replicate test-time adaptation benchmarks CIFAR10-to-CIFAR10-C, CIFAR100-to-CIFAR100-C, and ImageNet-to-ImageNet-C (inspired by Wang et. al), noting test error of each variation. Our end deliverables will be:

- A repository with reproducible code of our experiment.
- A paper detailing our methods and results.