Multimodal Learning on Temporal Data

Ye Xue[1] , **Diego Klabjan**[2] and **Jean Utke**[3]
[1,2]Northwestern Universion
[3]Allstate Insurance Company
yexue2015@u.northwestern.edu, d-klabjan@northwestern.edu, jutke@allstate.com

Abstract

In recent years, multimodal learning has attracted an increasing interest. A special scenario of multimodal learning, multimodal learning on temporal data, is common but has not been well studied. In multimodal temporal data, not all modalities of a sample arrive at the same time. Because of that, different types of samples may have different importance in many use cases, where an early sample with significant modalities may be more valuable than a later one as early predictions can be made to speed up decision making processes. Besides, sample correlations are very common in multimodal temporal data, as samples accumulate in time and a late sample may contain the same data existing in an earlier sample. Training without the awareness of the importance and correlation yields less effective models. In this work, we define multimodal temporal data, discuss key challenges and propose two methods that improve traditional multimodal training on such data. We demonstrate the effectiveness of the proposed methods on several multimodal temporal datasets, where they show 1% to 3% improvements over the baseline.

1 Introduction

Deep learning is enjoying great success in the last decade. It tackles numerous challenging tasks on unstructured data, such as images and text, and enables people to build astonishing applications that rely on deep learning techniques. However, each technique usually solves tasks with one modality. The Convolution Neural Network (CNN) is shown to be very effective on image tasks and the Recurrent Neural Network (RNN) and transformer [Vaswani *et al.*, 2017] are commonly used in language tasks. Although some recent works use CNN in language tasks and transformer on image tasks [Dosovitskiy *et al.*, 2021], the tasks considered are still single-modal tasks.

In recent years, multimodal learning is attracting an increasing interest. As computer vision and language models are advancing rapidly, multimodal models start to provide break through improvements, which unlocks new applications that naturally involve two or more modalities. From the healthcare field where fusing medical images and electronic health records shows improvements in performances when compared to models that used only single modalities [Huang *et al.*, 2020; Li *et al.*, 2021], to autonomous driving where intelligent systems are built to process various signals in different modalities [Feng *et al.*, 2020; Prakash *et al.*, 2021].

Multimodal temporal data are a special type of multimodal data very common in many real-world applications. Each modality of a particular sample can arrive or become available at a different time. For example, in-hospital patients usually do not take all tests at the same time. Vitals and demographics are usually available at an early stage of admission. Then various lab tests may be taken during the admission. If further analyses are needed, medical images like X-rays and text data such as bedside notes may finally become available. Although lab tests needed for each patient can be different, a common sequence of critical tests for a particular cohort may still be defined, e.g., common tests that help identify stages of cirrhosis [Bonekamp *et al.*, 2009]. A multimodal temporal model should be trained on such data to assist physicians' decision making.

More formally, we assume modalities sparsely arrive over a total of T time steps in a fixed order $(M_0, M_1, ..., M_T)$. Let $\mathcal{T} = \{0, 1, ..., T\}$. Multimodal temporal data consist of instances where each instance contains several multimodal samples. An instance accumulates samples over time as new modalities become available. A modality can be missing at a certain time (sparse arrival). An instance $\mathbf{X}^i$ is constructed from $\mathbf{x}^i_t = (\bar{\mathbf{x}}^i_1, \bar{\mathbf{x}}^i_2, ..., \bar{\mathbf{x}}^i_t)$, where $\bar{\mathbf{x}}^i_j$, $1 \leq j \leq t$ is the feature vector corresponding to modality M_j of instance i. If a certain modality M_j has not been received at time j (represented as $\square$), then $\bar{\mathbf{x}}^i_j = null_j$. Here $null_j$ is a fixed vector representing modality j (and being significantly different from all feature vectors of modality j). We call vector $\mathbf{x}^i_t$ a multimodal sample and an instance is $\mathbf{X}^i = (\mathbf{x}^i_t | t \in \mathcal{T})$. A multimodel temporal dataset consisting of N instances is then denoted as $\mathcal{D} = \{(\mathbf{X}^i, y^i) | i = 1, 2, ..., N\}$ with y_i being the ground truth. In the multimodal temporal data problem, we assume samples of all time steps are available in training and the goal is to train a model that can make good predictions at any time step given multimodal samples of various modalities.

The multimodal temporal scenario resembles the case of online learning; new data arrive in the sequential order. However, in multimodal temporal data, it is the modalities of each sample that arrive in a sequential order. Meanwhile, the instances may also arrive in a sequential order, for example, when new patients register. The latter is the case that online learning addresses, but in this work, we focus on the former case, which is the core characteristic of the multimodal temporal data.

There are a few challenges to learning multimodal temporal data. First, there are possible missing modalities. It requires a model that can handle inputs consisting of a various sets of modalities. Omninet [Pramanik *et al.*, 2019] has been proposed recently to solve this challenge. Second, data that are available at an earlier time may be more valuable if we can get some useful signals for early predictions. Failure to capture this may result in a model that does not consider correct attention and performs badly on valuable inputs. Third, samples are correlated, as a late sample contains the same data as an earlier sample. In such a case, a model may tend to be overfitted on duplicated modalities.

In this work, we focus on the last two challenges. We propose a modality weight optimization method to address the second challenge where some modalities are more valuable than others. Each modality is associated with a weight during training. The weights are embedded into the loss function and guide training so that the model is aware of the importance of each modality. We use Bayesian Optimization to find a good set of weights.

We tackle the third challenge of sample correlation by breaking down the original temporal data into groups ordered by time. The model is trained on one group at a time. By isolating samples by time, we break the correlation introduced by the temporal aspect of the data. However, this introduces another issue, catastrophic forgetting, where the model's performance on an old group decreases after being trained on a new group. We mitigate this issue by restricting the changes on model weights with Elastic Weight Consolidation (EWC) [Kirkpatrick *et al.*, 2017].

We evaluate proposed methods by comparing them to the traditional multimodal multitask training on several multimodal temporal datasets, which cover a wide range of modalities, such as images, text, videos, audio recordings and time series. The proposed methods show significant improvements on all datasets. Our contributions are summarized as follows.

1. We define multimodal temporal data and study key challenges in training with such data.

2. Two methods are proposed to address these challenges with code made public at http://xxx.

3. We conduct experiments on several multimodal temporal datasets and show that the proposed methods can improve standard multimodal multitask training by 1% to 3%.

In Section 2, we discuss related work. The problem of multimodal temporal data is defined in Section 3. We describe our methods in Section 4. The datasets and experimental setup are described in Section 5. Section 6 discusses the computational results and the conclusions are drawn in Section 7.

2 Related Work

A similar area to multimodal learning on temporal data is multimodal time-series learning, also known as temporal multimodal learning [Yang *et al.*, 2017; Schockaert, 2020]. Different from traditional time series, multimodal time series may contain multiple series of different modalities, such as audio-visual data. An RNN or transformer-based model is usually used with a fusion mechanism to learn a joint representation of multiple modalities. The difference of multimodal temporal data is that, instead of having temporal features in each modality, multimodal temporal data have modalities arranged in time.

In traditional continual learning settings, tasks are usually composed of data of the same modality. It has been recently extended to the multimodal scenario. Sun et al. [2020] propose a multimodal continual learning framework and a method to jointly update feature vectors of multiple modalities. Although data in each task can belong to a different modality, one task is limited to one modality, which is not applicable in our multimodal temporal setting, where each task may involve multiple modalities.

The sample correlation issue introduced by duplicating modality data is different from the temporal autocorrelation in time series analysis [Stojanova, 2013], where the previous data point might suggest the likelihood of the next data point. The modalities in the multimodal temporal data are not necessarily correlated. The sample correlation in our case is similar to the data duplication issue, where sample removal or downsampling are often used to handle duplication. But in our case, a sample might only be partially duplicated with others and any removing or downsampling causes data loss.

Another similar case of sample correlation is seen in [Lu *et al.*, 2020]. In training a model on multiple vision-and-language datasets together, some samples in one dataset are also present in whole or partially in other datasets. Although in this work the authors do not model overlapping of modalities, a dynamic stop-and-go mechanism is proposed to mitigate the overfitting problem, which might also be an issue in our case due to sample duplication. Since early samples may be duplicated in later samples multiple times, the model can be quickly overfitted on some early samples. The dynamic stop-and-go mechanism does not work well in our case as the duplication is much more prevalent. Stopping training the model on those early samples can not directly prevent overfitting as the model will also be trained and get overfitted on their duplicates in later samples.

3 Problem Definition

In the multimodal temporal data problem, we assume samples across all time steps are available in training and define the set containing training, validation and test data as $\mathcal{D} = \{(\mathbf{X}^i, y^i) | i = 1, 2, ..., N\}$. In inference, predictions can be made at any time, i.e., for any sample. The prediction at a later time is usually better than at an earlier time as the model makes a prediction with potentially more modalities. But the predictions at an earlier time are usually more valuable from the usage perspective, e.g. early medical diagnosis. Additionally, modalities are of different importance in terms of quality, difficulty of acquisition, etc. Depending on actual use cases, a model that performs in isolation better on important modalities and worse on less important ones may be more valuable than a model performing in the opposite way. In order to capture this, we define a weighted metric across all types of samples with different modalities. The goal is to train a model that has the best overall performance on all types of samples.

To define this metric, we first need to define types of samples. The type of a sample $\mathbf{x}_t^i$ is defined by its sparsity pattern which because of the assumed fixed modality order we can call the *modality type*

$$\mathbf{m}_t^i = (e_1^i, ..., e_t^i), e_j^i = \begin{cases} 1 & \bar{\mathbf{x}}_j^i \neq null_j \\ 0 & \bar{\mathbf{x}}_j^i = null_j. \end{cases}$$

All possible modality types of a multimodal temporal dataset are defined as $\mathcal{M} = \{\mathbf{m}_t^i\}_{i=1,2,...,N,t\in\mathcal{T}}$. The model gets different scores when making correct predictions on different modality types. We define types based on modalities instead of time because the time-based typing does not consider the importance of modalities. For example, at the same time $t = 3$, a sample may have modalities (video, image, text) and another sparse sample may have modalities (video, □, text). We may want to give more credit to the model if it makes a correct prediction on the latter sample with less available modalities. The time-based typing would not capture this. The time-based typing is also a special case of the modality-based typing when there are no missing modalities.

The model gets a reward based on its performance on samples of a certain modality type. Rewards are assigned based on reward functions, which are pre-defined by users of the model to quantify how certain modality type is valued. We denote a reward function as $r_m(v(\mathcal{D}^{(m)}))$, where $m \in \mathcal{M}$ is the modality type and $v(\mathcal{D}^{(m)})$ is the value of a performance metric, such as accuracy, F1 score, etc., on the subset $\mathcal{D}^{(m)}$ of samples in $\mathcal{D}$ whose type is m. We use accuracy as an example in later discussions for convenience. The function provides a reward for a model when it achieves a certain accuracy on a particular modality type during the evaluation. The overall performance of a model is the total reward of all types: $R = \sum_{m\in\mathcal{M}} r_m(v(\mathcal{D}^{(m)}))$.

The reward function should satisfy the following characteristics: it is an increasing function of accuracy given a particular modality type and it is an increasing function of the importance of a modality type given accuracy. In this work, we use an exponential function $r_m(v) = a_m \cdot v^d$ as the base of the reward function, where v represents accuracy. A more important modality type m is associated with a higher coefficient a_m. The coefficient d determines the increasing emphasis of rewards. The values of these coefficients are chosen given a particular use case, which are introduced later in Section 5.4.

4 Methodology

4.1 Background

Base Model

We use Omninet [Pramanik *et al.*, 2019] to learn multimodal temporal data. It is a transformer-based architecture that can handle inputs in various modalities, thanks to the design of temporal and spatial caches. Omninet encodes inputs by learning the temporal and spatial representations and stores them in caches. Each cache is a sequence of encodings, which may come from different modalities. With caches, Omninet can also handle inputs with missing modalities. Without such property, the multimodal temporal data with potential missing modalities can hardly be modeled effectively.

Bayesian Optimization

Bayesian optimization is a technique for optimization of black-box functions. In this work, we treat the base model trained on all samples with given loss function weights as a black-box function and use Bayesian optimization to find the best reward. Vanilla Bayesian optimization methods are typically computationally expensive. In this work, we use a more efficient method that combines **B**ayesian **O**ptimization and **H**yper**B**and (BOHB) [Falkner *et al.*, 2018]. Hyperband [Li *et al.*, 2017] is a bandit strategy that dynamically allocates resources to a set of configurations and uses successive halving [Jamieson and Talwalkar, 2016] to stop poorly performing configurations. Essentially, BOHB first tries multiple configurations with a lower budget (e.g., by stopping a training process early). Then it discards bad runs and continues training only on the promising runs.

Continual learning

Continual learning is a concept to learn a model on a sequence of tasks [Chen and Liu, 2018]. A common issue in continual learning is catastrophic forgetting, where the model forgets the knowledge learned from previous tasks after being trained on new tasks. Elastic Weight Consolidation (EWC) [Kirkpatrick *et al.*, 2017] is a well-known method to effectively handle the catastrophic forgetting issue. We leverage this technique in our method to mitigate the catastrophic forgetting issue when training on multimodal temporal data with the idea of continual learning. Different from standard continual learning, where each task is trained once, we are not limited by such a setting and train each task multiple times.

4.2 Modality Weight Optimization

Since different modality types are weighted differently in testing according to reward functions, ideally the model should be trained with such information as well. Without the awareness of such information, the model would not necessarily pay more attention to important modalities. As a result, the model might perform "equally" well on all modality types or might even perform worse on more valuable modality types, which can all yield a subpar performance of the total reward on all modality types.

To solve this issue, we propose a Modality Weight Optimization (MWO) method, where we assign weights $\boldsymbol{w}$ to training samples to capture the impact of reward functions and optimize the model $\boldsymbol{\theta}$ by the following loss function:

$$\begin{aligned} \hat{\boldsymbol{\theta}}(w) &= \arg\min_{\boldsymbol{\theta}} L(\boldsymbol{\theta}|\boldsymbol{w}; \mathcal{D}) \\ &= \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} \sum_{t=1}^{T} \boldsymbol{w_{i,t}} l(\mathbf{x}_t^i, y^i; \boldsymbol{\theta}). \end{aligned}$$

Here l can be any loss function approximating v, e.g., cross-entropy if the metric of interest is accuracy. Assigning a unique weight for each sample is intractable as it introduces too many parameters to optimize, considering that the training set is usually very large. Instead, we allow samples to share weights. For example, the same weight may be shared among samples with the same modality type or with other characteristics, according to a specific use case. Therefore, we also define a multiset function $\Delta : \boldsymbol{w}' \rightarrow \boldsymbol{w}$, where $\boldsymbol{w}'$ is a smaller set of shared weights.

Assigning weights to best reflect the impact of reward functions is not trivial. Therefore, instead of manually assigning weights, we use Bayesian Optimization to learn the best weights, which optimize the total reward. The objective function is defined as:

$$J(\boldsymbol{w}') = \sum_{m \in \mathcal{M}} r_m(v(\mathcal{D}^{(m)}; \hat{\boldsymbol{\theta}}(\Delta(\boldsymbol{w}')))).$$

We stress that the metric of interest is evaluated on the test data subset of $\mathcal{D}^{(m)}$ and the predictions depend on optimal training weights $\hat{\boldsymbol{\theta}}$.

4.3 Multimodal Continual Learning (MMCL)

Another challenge in the multimodal temporal problem is that samples are correlated. The nature of the growth of instances results in data duplication. In an instance, the data arriving in the first time step also appear in other samples at a later time step. Training with these correlated samples may cause overfitting. Since later samples contain the same modalities that appeared also in early samples, the model can be easily overfitted on early samples due to duplication. Simply pausing training on overfitted samples does not work well on the multimodal temporal data because other samples may still contain duplicates.

To tackle this issue, we propose **mu**lti**m**odal **c**ontinual **l**earning (MMCL). The idea is to break down the whole data with a mix of all modality types into groups in the order of time. Each group contains a smaller set of modality types. We denote all possible modality types of the group at time t as $\mathcal{M}_t = \{\mathbf{m}_t^i | i = 1, 2, ..., N\}$. The groups are arranged in the natural temporal order of modalities.

The model is trained on groups one by one in the time order. When training on M_t, the training data consist of $\mathcal{D}_t = \{(\mathbf{x}_t^i, y^i)\}_i$. To handle the catastrophic forgetting issue and preserve the model's performance on tasks in old groups, we train the model by adding an EWC loss to all older groups within a window. The window size is set to $T - 1$, that is, we add an EWC loss to the model of the current group from each of the other groups. After the model is trained on all groups once, one pass of training is finished. We continue training again starting from the first group. Multiple passes of training are performed. We index each group i based on its pass p and time step t as $i = (p - 1) \cdot T + t$. The group i is trained with the following loss:

$$L(\theta_i) = l(\theta_i) + \sum_{j=\max(i-T+1,1)}^{i-1} \lambda_{i,j} \cdot \Psi(\theta_i, \hat{\theta}_j, F_j),$$

Algorithm 1 MMCL

θ_1 are the initial model weights; λ_0 is the base value of the EWC constraint; α, β and γ are hyper-parameters for scheduling λ.

```
 1: initialize $\theta_1$
 2: for each pass $p = 1, 2, ..., P$ do
 3:     for $t = 1, 2, ..., T$ do
 4:         $i \leftarrow (p-1) \cdot T + t$ // current group index
 5:         $k \leftarrow \max(i - T + 1, 1)$ // oldest group index
 6:         for $j = k, k+1, ..., i-1$ do
 7:             if $i < j$ or $i \bmod T = j$ then
 8:                 $\lambda_{i,j} \leftarrow 0$
 9:             else
10:                 $\lambda_{i,j} \leftarrow \begin{cases} \textbf{option 1 } \lambda_0 \\ \textbf{option 2 } \frac{\alpha^{j \bmod T}}{\gamma^{\lfloor i/T \rfloor} \beta^{i \bmod T}} \lambda_0. \end{cases}$
11:             end if
12:         end for
13:         $\mathcal{D}_t \leftarrow$ data with modality type $\in \mathcal{M}_t$
14:         $\hat{\theta}_i = \min_{\theta_i}(l(\theta_i; \mathcal{D}_t) + \sum_{j=k}^{i-1} \lambda_{i,j} \cdot \Psi(\theta_i, \hat{\theta}_j, F_j))$
15:         $F_i \leftarrow$ UpdateFisher($\hat{\theta}_i, \mathcal{D}_t$)
16:         $\theta_{i+1} \leftarrow \hat{\theta}_i$
17:     end for
18: end for
```

where $\Psi(\theta_i, \hat{\theta}_j, F_j) = \sum_p F_{j,p}(\theta_{i,p} - \hat{\theta}_{j,p})^2$ is the EWC loss on all parameter p and $\lambda_{i,j}$ is the weight between the current group i and older group j. After training the model on a group, we update the Fisher matrix F for this group using the UpdateFisher function, same as [Kirkpatrick *et al.*, 2017]. The full training process is shown in Algorithm 1.

The hyper-parameters λ control forgetting. A larger value of λ's implies a stronger limitation of the model weight changes so the weights stay within the low-loss manifold of the older groups. The most straightforward way of specifying λ is to use a constant value, as shown in Algorithm 1, line 7 option 1. However, it poses several limits. For example, it implies that each group is equally important. When training the model on an important group, we may want to apply less emphasis from old groups, so the model has more freedom to explore the model weight space to find the optimum for the current group. When training the model on a less important group, we want to apply a large weight from important old groups, so that the model can preserve its performance on those important groups.

Considering that the importance of a sample decreases as the time step increases, we propose a scheduling scheme of λ as follows:

$$\lambda_{i,j} = \begin{cases} 0 & (i \bmod T) = j \text{ or } i < j \\ \frac{\alpha^{j \bmod T}}{\gamma^{\lfloor i/T \rfloor} \beta^{i \bmod T}} \lambda_0 & i > j. \end{cases}$$

where $\alpha, \beta, \gamma \in (0, 1)$. Value α assumes that an earlier group receives a higher importance than a later group. Parameter β imposes an increasing trend during training within each pass. As we are training on less and less important groups within a pass, we increase the model's ability in preserving its per-

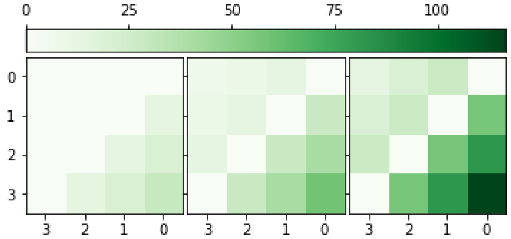


Figure 1: Example of a schedule of λ for 3 passes of training. In each of the 3 heat maps, the y-axis shows the time steps of the current groups and on the x-axis are the time steps of old groups. We reverse the axis for better visualization. We have white blocks on diagonals because we do not apply weights on the current model itself. These heat maps show the following patterns. (1) In the same pass, given a current group, the value of λ increases as the time step of an old group decreases (left → right); (2) In the same pass, given an old group, the value of λ on it increases as the training goes on (top → down); (3) Given a current and an old group, the value of λ increases as the training pass increases (left → right).

formance on old groups by increasing the constraint. Finally, γ makes an increasing trend of constraints pass by pass, to enhance the performance preserving as the model approaches an optimum. Note that we separate the control of the intra-pass increasing by β and inter-pass increasing by γ. This introduces more flexibility in scheduling λ compared with a uniform increasing trend controlled by one parameter. For example, because of β, at the end of each pass, the emphasis on each group reaches a very high value. The model is considered in a "preserving" mode. Starting the next pass, especially in early passes, we may want to encourage the model to restart "exploring." This can be achieved by properly setting γ. Figure 1 shows the heat map of an example schedule of λ.

5 Datasets

5.1 BDD Multimodal Temporal Dataset

The Berkeley DeepDrive Video dataset [Yu *et al.*, 2020] is a dataset consisting of real driving videos (V) and GPS/IMU (G) records. Similar to [Xu *et al.*, 2016], we train an action prediction model on this dataset. We consider 4 discrete driving actions: straight, stop, left turn and right turn. Each video is truncated to 16s and temporally downsampled to 3Hz to avoid duplicate frames.

We define an order of modality for this dataset as (G,G,G,V,G,V,V,V) to simulate a use case where GPS signals usually arrive earlier than videos because GPS signals are smaller. Then we convert it to a multimodal temporal dataset (BDD), where data of each modality are 2s in length. We assume a modality always arrives at the corresponding time step. The BDD dataset in total contains 8 modality types. We use the same splits of training, validation and test set as provided in the original data, and convert each split into a timeline dataset. This also applies to other datasets introduced in the following sections. In the BDD dataset, there are 154,696 training, 14,936 validation and 14,936 test samples.

Table 1: Test reward comparison

Datasets	Baseline	MWO	MMCL
BDD	0.3665	+3.18%	+2.40%
SIQ	0.1781	+1.27%	+1.12%
ITV	4.6317	+1.32%	+0.95%

5.2 Image-Text-Video Dataset

We create the **I**mage-**T**ext-and-**V**ideo (ITV) multimodal temporal classification dataset from three common public datasets. The image data come from the Cifar10 dataset [Krizhevsky *et al.*, 2009]. The original Cifar10 contains 10 classes. We group animals into one class and others into another class. We balance two classes by moving images of birds to another class. Text data come from the IMDB dataset [Maas *et al.*, 2011], which is a binary sentiment classification dataset containing movie reviews that are labeled either positive or negative. Videos come from the HMDB dataset [Kuehne *et al.*, 2011], which is a large human motion recognition dataset with 51 actions. Actions are grouped into 5 types: general facial actions, facial actions with object manipulation, general body movements, body movements with object interaction and body movements for human interaction. We further group the 5 types into two classes. One class contains general body movements and body movements for human interactions and the other contains the rest of the types.

We assign binary labels to the modified source datasets and consider samples with the same label belonging to the same class. That is, we treat images, text and videos of class 0 as one class and the rest as another class.

For this dataset, we define an order of modality as (I,T,T,I,V). To create an instance, we draw a sample for each modality from the corresponding source dataset. To simulate a more general case, we introduce missing modalities by sampling with a missing rate of 30% or higher. Since the three source datasets differ in size, in order to utilize as many samples from all sources, we adjust the missing rate for each modality. Modalities from a smaller source dataset have a higher missing rate. The final ITV dataset contains 17 modality types due to missing modalities. There are 61,070/ 19,166/19,030 training/validation/test samples.

5.3 Social-IQ Dataset

Social-IQ [Zadeh *et al.*, 2019] is a video question answering dataset that contains 1,250 annotated videos, 7,500 questions and 52,500 answers. Each question is provided with 4 correct answers and 3 incorrect answers. All answers are sentences. The task is to predict whether an answer is correct given a video with a question. We break down each video into visual, language and acoustic modalities. The visual modality contains video frames extracted at 1fps. On average, each video sample consists of 55 frames. Transcripts are the input of the language modality. The acoustic modality contains audio features, which are extracted from COVAREP [Degottex *et al.*, 2014]. They are pitch and frequency-related features and provide different signals from transcript features.

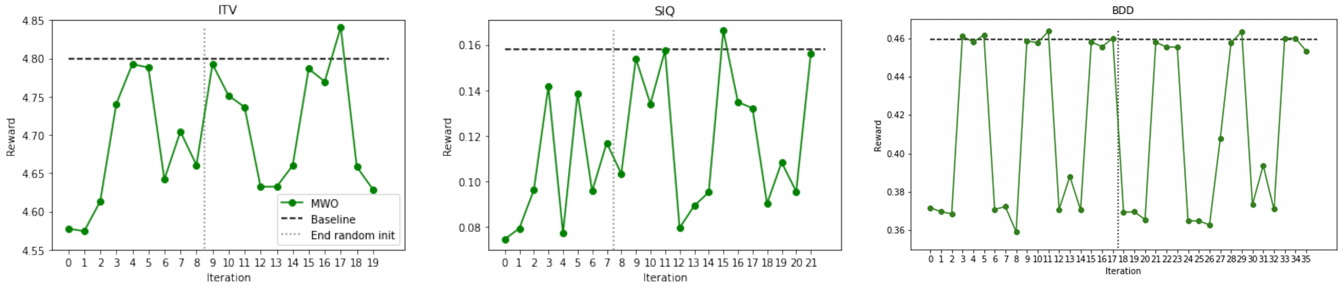


Figure 2: Reward curves on validation set

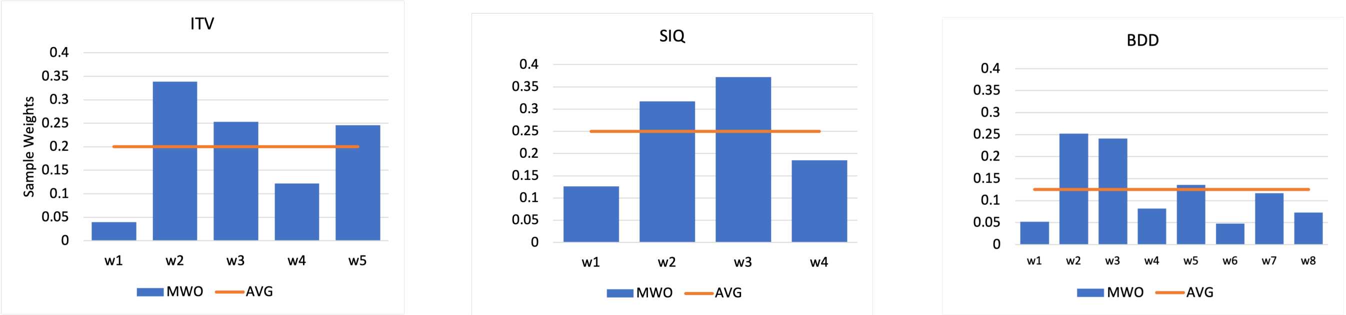


Figure 3: Best modality weights

We turn the original Social-IQ into a multimodal temporal dataset, SIQ, by arranging inputs in the following order: questions (Q), audios (A), transcripts (T) and frames (V). We arrange them in this order to simulate a use case where a later modality provides more useful signals. We quantify the usefulness of a modality by accuracy. The authors of Social-IQ show an accuracy of 57.02, 57.22, 57.87 and 63.91 on questions, audios, transcripts and frames, respectively. The SIQ dataset has 138,040 training, 16,884 validation and 16,884 test samples.

5.4 Reward Functions

For the BDD and SIQ dataset, we define reward functions as $r_t(v) = 0.7^t \cdot v^8$. In these two datasets, we simulate the scenario that the importance of an input decreases in time. Also, since we assume there is no missing modality in these datasets, the time-based reward function is the same as modality-based reward function. For simplicity, we define the reward function over time instead of modality. The coefficients are set to these values so that for all modality types, the change in reward is reasonably large when accuracy changes. The same logic also applies to the reward function of the other dataset.

In the ITV dataset, we simulate a different scenario where a sample with more available modalities has a lower reward. We do not use the same reward function above. Usually, a sample in a later time has more available modalities than an earlier sample. However, this is not always the case when there are missing modalities. For example, a sample at time step 5 of instance A may only have 1 available modality due to missing modalities, while a sample at time step 2 of instance B may have 2 available modalities. The reward function for this dataset is $r_m(v) = g(m) \cdot v^8$, where $g(m) = 1 - \frac{|m|-1}{T}$. Function $g(m)$ is set in such a way so that it decreases linearly in $[1, 0)$ as $|m|$, the number of modalities, increases. Instead of an exponential modality multiplier 0.7^t used in the other two datasets, we use a linear function for the multiplier here. We cannot experiment with all types of functions and choose these two common types of functions, so that our experimental results are more general.

6 Results

Our algorithms are implemented in PyTorch. For BOHB, we use its official implementation[1]. All experiments are conducted on NVIDIA GeForce RTX 2080 Ti GPUs. We use Omninet as the base model and compare proposed methods, MWO and MMCL, against vanilla multimodal training as the baseline.

We tune learning rates for all models by grid search. For MWO, we do not tune additional hyper-parameters in the BOHB algorithm and use the default values in its official library. For MMCL, we find the best scheduling scheme of λ by tuning α, β and γ. We empirically set λ_0 to 10 with the observation that the model barely changes when $\lambda > 300$. We thus obtain a scheduling scheme in $(0, 300]$. Even with a different λ_0, we were still able to find a similar scheduling scheme by manipulating α, β and γ. Therefore, we focus on tuning these three hyper-parameters. We tune them on grids $(0.5, 0.6, 0.7, 0.8)$ with the following heuristics, considering the large number of combinations. We randomly try some combinations, then fix one hyper-parameter of the best com-

[1] https://github.com/automl/HpBandSter

bination and tune the other two using grid search. The model may achieve a better performance with more tuning, but the current tuning already yields many strong models.

We tune models on the validation data and select the best based on their validation rewards. We train the best model 3 times with different random seeds and record the average reward on the test data. The test results are shown in Table 1. It shows the absolute test results of the baseline model and the relative changes of our models over the baseline. Both proposed methods outperform the baseline on all datasets. The improvement ranges from 1% to 3%.

6.1 Modality Weight Optimization

Figure 2 shows the validation reward curves of the MWO model on all datasets. The best validation reward of the baseline model is shown as the horizontal dashed line. Each dot in the plot is a result of training Omninet with specific modality weights. The BOHB kernel needs some initial data before making reasonable predictions [Falkner *et al.*, 2018]. We mark the end of the random initialization stage as a vertical line. In the random initialization stage, Omninet is trained with random modality weights. After that, it is trained with weights proposed by the BOHB kernel. The number of randomly initialized configurations depends on the size of the parameters (i.e., modality weights) to optimize [Falkner *et al.*, 2018].

Since BOHB allows us to try more configurations with a small budget, in the plots we see two groups of dots. One group contains configurations running with the full budget and the other is from those running with a smaller budget. Budgets are defined in terms of training epochs. We determine the full budget by recording the number of epochs that Omninet needs to achieve the best validation reward. We set the smaller budget to 1/3 of the full budget, as we observe that this is in many cases enough to determine whether a configuration is a good one or not. We empirically limit the number of BOHB iterations to 15 times the full budget for the ITV and SIQ dataset, and 25 times for the BDD dataset. Although running more iterations might further improve the performance, we observe that within these iterations the model can already provide good improvements.

Figure 3 shows the best modality weights. Modality weights are normalized and plotted with a line indicating the average of all weights. Although the reward function imposes a decreasing importance score in the time order or as more modalities are available, we do not see a clear decreasing trend of the weights. We argue that the importance of the modality itself on the target also plays a role. The model may want to pay more attention to a particular modality that contributes more to the overall performance regardless of the small weight we put on it during evaluation. Interestingly, the first modality always receives a very low weight. This could be due to the fact that samples of the first modality are duplicated the most, since later samples always contain the available samples from previous time steps. Assigning a low weight help the model be less biased towards those duplicated samples. The learned modality weights are a result of balancing between the multiple aforementioned factors.

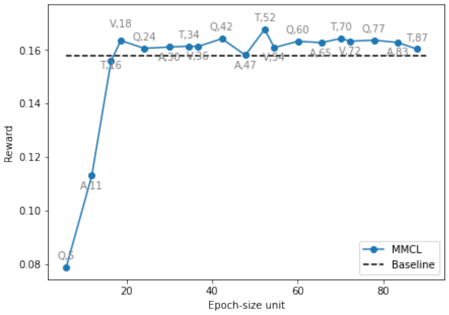


Figure 4: MMCL reward

MWO shows a 3.18% reward gain on the BDD dataset, 1.27% on the SIQ dataset and 1.32% on the ITV dataset. They are the test results of the best models, which are selected based on validation rewards shown in Figure 2. Interestingly, the performance gap between our model and the baseline increases as the number of time steps increases. For datasets with a longer order of sequence, it is harder to balance the weight of each modality type to achieve the optimal reward. This does provide more opportunities for our model to improve over the baseline. However, it also increases the complexity of the problem and usually requires more iterations to find a good solution.

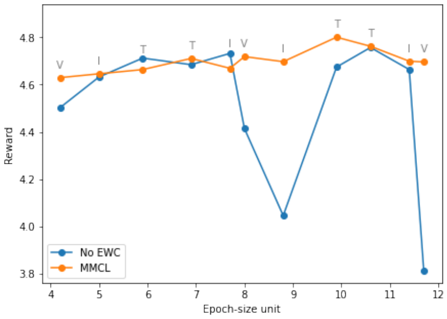


Figure 5: MMCL vs. a training without EWC

6.2 Continuous Multimodal Temporal Training

MMCL improves the baseline by 2.4% on the BDD dataset, 1.12% on the SIQ dataset and 0.95% on the ITV dataset. Figure 4 shows the MMCL reward curve on the SIQ dataset. After training on each group, we record the total reward calculated on all groups in the validation set. Since we break the data into multiple groups and train one after the other, an epoch has a different definition between MMCL and the baseline. Instead, we use an epoch-size unit, where in one unit the model is trained on the same number of samples as one epoch in training the baseline. In the plot, we name the group with the corresponding modality that arrives at that time step. More visualizations are shown in Appendix A.

Lambda Scheduling

In MMCL, besides adding the EWC loss to preserve the model's performance on old groups, we also train the model on each group multiple times. An interesting question is whether the EWC loss is needed if the model can re-learn the knowledge on an old group. Figure 5 shows a comparison between MMCL and training without the EWC loss on the ITV dataset. We observe a very large variation of the total reward on the no-EWC training. For example, the performance drops dramatically around the 8th, 9th and 12th epoch on modality V, I and V, respectively. Without EWC, the model prioritizes the current group. The performance drop may happen when the current group contributes less to the reward, such as the group with modality V. The EWC loss stabilizes the model's performance across groups as the model always tries to preserve its performance on old groups.

Figure 1 shows an example of the best λ's used in the SIQ dataset. The values are calculated with $\alpha = 0.7$, $\beta = 0.7$ and $\gamma = 0.5$. We perform an ablation study on these three hyper-parameters. Each time, we disable one hyper-parameter by setting its value to 1. As shown in Figure 6, disabling any one of them yields a performance degradation. Disabling β or γ leads to more than 1% performance degradation, which indicates the importance of emphasizing important groups and enhancing the performance as the model approaches the optimum.

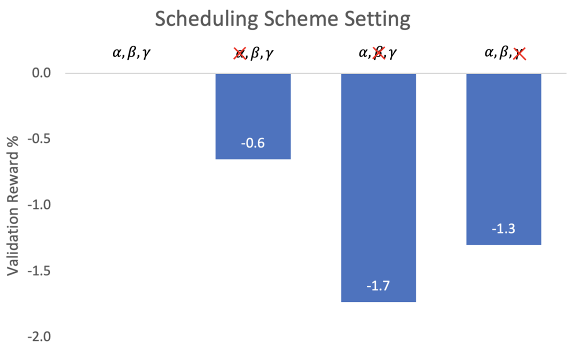


Figure 6: Ablation study for scheduling λ with (α, β,γ)

Restoring Learning Rate Factor

In our early experiments, we find that the model's performance stops improving after the first few groups, if we start a new optimizer for each group. The reason might be that the initial learning rate is too large for later groups after the model has been trained on a few groups. In our training procedure, the model may be trained on the same group multiple times. When it starts to be trained on the same data the second time, the model weights might be already close to the optimum and a small learning rate is expected. This problem is more severe considering that early samples are duplicated in the late samples. The model might need a smaller learning rate even before going into the second pass of training.

To overcome this issue, we propose a learning rate restoring strategy. In this work, we train Omninet with the Adam optimizer [Kingma and Ba, 2015] and schedule learning rate using the Noam scheduler [Shazeer and Stern, 2018] same as Pramanik *et al.*, [2019]. We adjust the learning rate by restoring the number of training steps from the previous group. We observe a 1.3% improvement on the ITV dataset with this strategy and also use it on the other two datasets. The reason we do not restore the optimizer states, i.e., momentum values, is that the data distribution of the next group is different from the previous one. Restoring the momentum values does not show improvements.

6.3 Discussions

The two proposed methods improve traditional multimodal training by tackling different problems in learning from multimodal temporal data. MWO shows a better performance than MMCL on all datasets. It may be because MWO benefits from directly optimizing the modality weights. MMCL also considers the importance of modalities but in a more implicit way by emphasizing important groups in EWC. However, MWO also takes a longer time to optimize. On average, MWO takes 5 times longer than the baseline to achieve the best validation reward and MMC only takes 1.2 times longer than the baseline. We recommend MWO when achieving the best performance is the top interest and suggest to use MMCL to balance computing resources and performance.

7 Future Work and Conclusion

Reinforcement learning on multimodal temporal data is an interesting direction, where an episode can be defined from the first modality to the last one and each time step corresponds to a state. Some challenges include but are not limited to: the states are not necessarily correlated with each other in multimodal temporal data and it is unclear how to define the environment and define interactions between the agent and the environment.

In this work, we define and study multimodal temporal data, which are common in real-world applications but have not yet been well studied. We discuss key challenges and propose a modality weight optimization method to address the "importance" challenge, where some samples may be more important than others. We propose a training algorithm with elastic weight consolidation to address the "correlation" challenge. The proposed methods demonstrate improvements over standard multimodal multitask training on several multimodal temporal datasets.

References

[Bonekamp *et al.*, 2009] Susanne Bonekamp, Ihab Kamel, Steven Solga, and Jeanne Clark. Can imaging modalities diagnose and stage hepatic fibrosis and cirrhosis accurately? *Journal of Hepatology*, 50(1):17–35, 2009.

[Chen and Liu, 2018] Zhiyuan Chen and Bing Liu. Lifelong machine learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 12(3):1–207, 2018.

[Degottex *et al.*, 2014] Gilles Degottex, John Kane, Thomas Drugman, Tuomo Raitio, and Stefan Scherer. COVAREP—a collaborative voice analysis repository for speech technologies. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 960–964, 2014.

[Dosovitskiy *et al.*, 2021] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *9th International Conference on Learning Representations*, 2021.

[Falkner *et al.*, 2018] Stefan Falkner, Aaron Klein, and Frank Hutter. BOHB: Robust and efficient hyperparameter optimization at scale. In *International Conference on Machine Learning*, pages 1437–1446, 2018.

[Feng *et al.*, 2020] Di Feng, Christian Haase-Schütz, Lars Rosenbaum, Heinz Hertlein, Claudius Glaeser, Fabian Timm, Werner Wiesbeck, and Klaus Dietmayer. Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges. *IEEE Transactions on Intelligent Transportation Systems*, 22(3):1341–1360, 2020.

[Huang *et al.*, 2020] Shih-Cheng Huang, Anuj Pareek, Saeed Seyyedi, Imon Banerjee, and Matthew P. Lungren. Fusion of medical imaging and electronic health records using deep learning: A systematic review and implementation guidelines. *NPJ Digital Medicine*, 3(1):1–9, 2020.

[Jamieson and Talwalkar, 2016] Kevin Jamieson and Ameet Talwalkar. Non-stochastic best arm identification and hyperparameter optimization. In *Artificial Intelligence and Statistics*, pages 240–248, 2016.

[Kingma and Ba, 2015] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations*, 2015.

[Kirkpatrick *et al.*, 2017] James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, and Agnieszka Grabska-Barwinska. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017.

[Krizhevsky *et al.*, 2009] Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. https://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.222.9220&rep=rep1&type=pdf, 2009.

[Kuehne *et al.*, 2011] Hildegard Kuehne, Hueihan Jhuang, Estíbaliz Garrote, Tomaso Poggio, and Thomas Serre. HMDB: A large video database for human motion recognition. In *International Conference on Computer Vision*, pages 2556–2563, 2011.

[Li *et al.*, 2017] Lisha Li, Kevin G. Jamieson, Giulia DeSalvo, Afshin Rostamizadeh, and Ameet Talwalkar. Hyperband: Bandit-based configuration evaluation for hyperparameter optimization. In *International Conference on Learning Representations (Poster)*, 2017.

[Li *et al.*, 2021] Yi Li, Junli Zhao, Zhihan Lv, and Zhenkuan Pan. Multimodal medical supervised image fusion method by CNN. *Frontiers in Neuroscience*, 15:303, 2021.

[Lu *et al.*, 2020] Jiasen Lu, Vedanuj Goswami, Marcus Rohrbach, Devi Parikh, and Stefan Lee. 12-in-1: Multi-task vision and language representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10437–10446, 2020.

[Maas *et al.*, 2011] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, 2011.

[Prakash *et al.*, 2021] Aditya Prakash, Kashyap Chitta, and Andreas Geiger. Multi-modal fusion transformer for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7077–7087, 2021.

[Pramanik *et al.*, 2019] Subhojeet Pramanik, Priyanka Agrawal, and Aman Hussain. Omninet: A unified architecture for multi-modal multi-task learning. *arXiv preprint arXiv:1907.07804*, 2019.

[Schockaert, 2020] Cedric Schockaert. A causal-based framework for multimodal multivariate time series validation enhanced by unsupervised deep learning as an enabler for industry 4.0. *arXiv preprint arXiv:2008.02171*, 2020.

[Shazeer and Stern, 2018] Noam Shazeer and Mitchell Stern. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pages 4596–4604, 2018.

[Stojanova, 2013] Daniela Stojanova. Considering autocorrelation in predictive models. *Informatica*, 37(1), 2013.

[Sun *et al.*, 2020] Fuchun Sun, Huaping Liu, Chao Yang, and Bin Fang. Multimodal continual learning using online dictionary updating. *IEEE Transactions on Cognitive and Developmental Systems*, 13(1):171–178, 2020.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

[Xu *et al.*, 2016] Huazhe Xu, Yang Gao, Fisher Yu, and Trevor Darrell. End-to-end learning of driving models from large-scale video datasets. *arXiv preprint arXiv:1612.01079*, 2016.

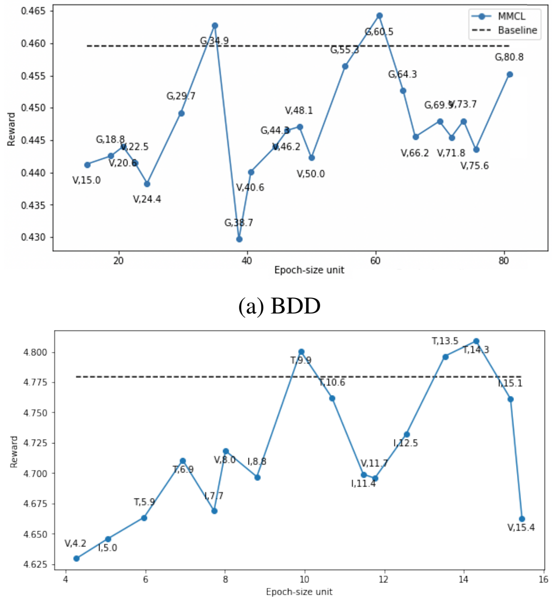
[Yang *et al.*, 2017] Xitong Yang, Palghat Ramesh, Radha Chitta, Sriganesh Madhvanath, Edgar A. Bernal, and Jiebo Luo. Deep multimodal representation learning from temporal data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5447–5455, 2017.

[Yu *et al.*, 2020] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. BDD100K: A diverse driving dataset for heterogeneous multitask learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2636–2645, 2020.

[Zadeh *et al.*, 2019] Amir Zadeh, Michael Chan, Paul Pu Liang, Edmund Tong, and Louis-Philippe Morency.

Social-IQ: A question answering benchmark for artificial social intelligence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8807–8817, 2019.

A MMCL Training Curves



(a) BDD

(b) ITV

Figure 7: MMCL validation rewards

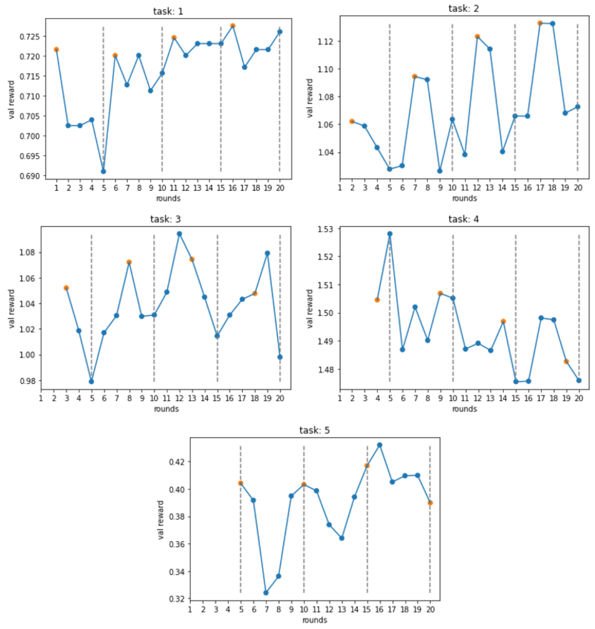


Figure 8: MMCL reward by groups

Figure 7 shows the validation reward curves of MMCL on the BDD and ITV datasets. In Figure 8, we plot the validation reward by tasks. Each task corresponds to training on a particular group and the index of the task is the time step of the given group. We plot the training on the ITV dataset as an example. In each plot of a task, the reward is the summation of rewards on all modality types that only exist in this group. For example, the ITV dataset contains 5 modalities (I,T,T,I,V). Task 2 corresponds to a group of modality types, including (0,1,0,0,0) and (1,1,0,0,0); task 3 corresponds to another group of modality types, including (1,1,1,0,0), (0,0,1,0,0), etc.

It shows the model's performance on a given group during the whole training process. The orange dots in each plot show the performance of the model after it has been trained on the corresponding group. The vertical dashed lines mark the end of each training pass. We observe that the model's performance on a particular group usually drops after being trained on new groups. This is a typical pattern in continual learning. However, we observe that the model's performance bounces back after being trained again on the same group. More importantly, the model's performance generally improves after each subsequent encounter (i.e., the model's performance on a certain group improves over passes), especially in early passes, which also demonstrates the need of training on the same group multiple times.