Arbitrary Predictive Coding for Improving Action Classification Accuracy

Tamique de Brito  
Massachusetts Institute of Technology  
tamique@mit.edu

Dr. Yogesh Rawat  
University of Central Florida  
yogesh@crcv.ucf.edu

Dr. Mubarak Shah  
University of Central Florida  
shah@crcv.ucf.edu

July 29, 2020

Abstract

Predictive coding is an approach that involves making predictions in latent space. Some work has been done in using predictive coding to learn self-supervised video features [1, 2]. This paper explores a variant of predictive coding, which we call “arbitrary predictive coding”, in both semi-supervised and fully-supervised settings. In addition, we present a loss function, which we call “partitioned loss”, that explicitly utilizes the structure of the dataset we use, and has the effect of ‘factoring’ the latent representation according to the structure of the dataset.

We find that in a fully-supervised setting, jointly training classification with predictive coding improves classification accuracy. However, performing semi-supervised learning by using additional unlabelled data for predictive coding reduces classification accuracy.

Additionally, we find that applying the partitioned loss to the latent representations of examples provides an improvement to classification accuracy.

1 Introduction

Predictive coding has been shown to be successful in learning self-supervised representations for many different types of sequential data [1, 2]. These works have done prediction by applying an autoregressive model to latent representations at consecutive time-steps of sequential data, using the autoregressive model to predict latent embeddings of future time-steps.

In contrast, our approach takes latent representations from a few randomly-selected time-steps, along with temporal encodings (also called positional encodings) for the time-steps, aggregates them into a global representation, and uses the aggregated representation to predict latent embeddings at an arbitrary ‘target’ time-step given the time-step’s temporal encodings. Furthermore, the predictive coding combined with classification to form approaches for fully-supervised and semi-supervised learning. While this framework is general and could ostensibly be applied to any type of sequential data, we test it on the UCF-101 action recognition dataset [6].

There has also been recent work in using the structure of a dataset to learn (image) representations [4]. In the referenced work, the structure of a video dataset is learned to use visual invariances via predictive coding. In contrast, we do not perform predictive coding, instead enforcing similarity and contrastive constraints between examples in different parts of the dataset hierarchy. Furthermore, each similarity or contrastive constraint is applied to a different slice of the global representation, which has the effect of factoring the representation into components where each component represents is either invariant or variant to a particular (for example a component could be variant with different shots of the same video or invariant between different videos with the same label). See Section 2.1 for more details.

2 Framework

Our approach to predictive coding is illustrated in Figure 1b alongside the ordinary approach (Figure 1a) for comparison. The full proposed architecture is shown in Figure 2. For the encoder, we use an I3D network [5], applying a linear layer to the output of the final convolutional layer to reduce the latent dimension to 512. For the aggregator, a transformer encoder block [4] with one layer and four attention heads is used. An LSTM was also tried, but did not perform as well. The classifier head is a simple linear layer, and the predictor head is a two-layer feedforward network.

For a given video example, a set of five randomly-selected eight-frame clips are taken, along with temporal encodings that represent their positions in the video. Additionally a target clip to be predicted is selected, along with its temporal encodings. We use the temporal encodings described in [4] (section 3.5). The dimension of the temporal encoding is 512 to match with the latent dimension. The clips are then passed through the encoder, and the latent embeddings are added to the temporal encodings to incorporate the temporal information. The resulting embeddings are passed into the transformer encoder block and the output sequence is averaged to produce a global representation for the video. Then, the global representation is fed through the classifier head to produce a prediction of the class of the video. The global representation is also concatenated with the target temporal encoding and fed into the predictor head to predict the latent embedding of the target clip. Cross-entropy loss is used for classification, and MSE loss is used for prediction. The entire system is trained end-to-end.

In this architecture, there is no inherent limit on the number of clips that can be encoded and passed into the aggregator. Even though five clips are taken at train time, more can be used at evaluation time. As would be expected, this has been found to improve accuracy, usually by 1.5% or so. Some experiments
2.1 Partitioned Loss

Here, we provide a more detailed explanation of the partitioned loss. This loss utilizes the structure of the UCF-101 dataset. The structure of this dataset is illustrated in Figure 3. There are 101 classes at the highest level of hierarchy. Within each class, there are 25 videos. Each video is further broken down into several shots. The idea behind partitioned loss is that clips from different parts of the hierarchy have different relationships, and it should help to learn useful representations if, for each possible relationship, there is a component of the global representation that is invariant to the relationship, and a component that is variant with the relationship. An illustration of how the partitioned loss is applied is provided in Figure 4. How the partitioned loss fits into the joint training architecture is shown in Figure 6d.

3 Experiments

Three main experiments are conducted. The first consists of starting from only the I3D model and progressively adding components to see the improvement gained by each. The second consists of trying the approach with ImageNet-pretrained weights for I3D. The third experiment consists of testing effectiveness of the architecture for semi-supervised learning. There are two additional experiments. The first further evaluates the extent to which the model utilizes the temporal encodings, and the second evaluates the effectiveness of partitioned loss.

3.1 Main Approach

Here, the effect of progressively adding components is tested in a sequence of four sub-experiments. An illustration of the architecture for each sub-experiment is provided in Figure 6 in the Appendix. The first test in this sequence is a baseline, using only a single randomly-selected clip, which is passed through the encoder and then the classification head to produce a prediction (Figure 6a). Next, multiple clips are incorporated, encoding them all and passing them through the aggregator, to produce representation which is then used for classification (Figure 6b). No temporal encodings are used for
this experiment. Then, the multiple-clip approach is enhanced by adding temporal encodings, which allow the aggregator to utilize the temporal relationships between the clips (Figure 6c). Finally, the predictor head is added and trained jointly with the classification (Figure 2). This sequence of experiments is done using both 20% and 100% of training data. The results are shown in Table 1. It can be seen that each addition of a part to the model increases performance by some amount.

### 3.2 Pretrained Weights

Here, the effectiveness of our approach using pretrained weights is evaluated. We run the “multi-clip + temporal encodings” and “joint training” experiments using ImageNet-pretrained weights for I3D, using 100% of data for training. Here, a modification to the joint training was necessary to produce a result comparable to multi-clip + temporal encodings. Specifically, the weight given to the predictive coding needed to be slowly increased up to its maximum value, starting from zero. We call this approach “slow-start”. The results are shown in Table 2. Here, the joint training performs worse than multi-clip+temporal encodings, even with slow-start. This can be explained by the fact that the ImageNet-pretrained weights are optimized for static images and trying to learn to predict temporally-sensitive information from the start does not match with what the pretrained weights are providing. Before implementing the slow-start approach, the weights learned from multi-clip + temporal encodings were used as a starting point for the joint training, and this produced comparable results. As an additional note for comparison: an I3D by itself with ImageNet-pretrained weights obtains an accuracy of 84.5%, so the multi-clip + temporal encodings improves accuracy by 8.2%.

### 3.3 Semi-supervised Learning

Here, the effectiveness of our approach for semi-supervised learning is evaluated. The semi-supervised learning is done by running the joint training architecture on 20% of labelled training data, but adding additional unlabelled clips from remaining 80% of training data to each batch. The global representations for the unlabelled clips are fed into the predictor head, but not the classifier head. The results are shown in Table 3. The multi-clip + temporal encodings reference is simply copied from Table 2. However to obtain the fully-supervised reference, the semi-supervised architecture was rerun, but the unlabelled examples were substituted with clips from the already-labelled 20% of data. This was to ensure that the method of simply adding additional unsupervised clips to each batch would not have any significant side effects (as adding these clips might change the balance between classification and prediction). The decrease of accuracy with additional unlabelled data is surprising. On one hand, it suggests that the kind of features it learns for latent prediction are not useful classification; on the other hand, the fact that jointly training prediction with classification improves accuracy seems to show that these features are useful for classification. One reason that was initially suspected was that the use of MSE loss for prediction allowed for the learning of a constant representation for the unlabelled examples. However, even when MSE loss was substituted with a contrastive loss (specifically, triplet loss), this did not fix the problem.

### 3.4 Testing Utilization of Temporal Encodings

Here, the extent to which the model utilized temporal encodings is tested. A trained multi-clip + temporal encoding model is evaluated under the following conditions:

1. No temporal encodings are added to clip embeddings
2. Incorrect temporal encodings are added to embeddings
3. Correct temporal encodings are added to embeddings

In addition, a trained multi-clip model (without temporal encodings) is evaluated for comparison.

In each case, the model is evaluated for different numbers of input clips. Specifically, the model is evaluated for each of k=1, 2, 3..., 20 input clips. The results of this are shown in Figure 5. It can be seen that using the incorrect temporal encodings significantly impacts the models performance, thought not as much as not using the temporal encodings at all.

### 3.5 Testing Partitioned Loss

Here we evaluate the effect of partitioned loss on classification accuracy obtained. For this set of experiments, we use different components than in previous experiments. For the encoder we use a 3D convolutional network based off of C3D, and for the aggregator we use an LSTM. For the LSTM, temporal encodings are concatenated to clip embeddings instead of adding, as this was found to produce higher accuracy. The same sequence of experiments done in Section 3.1 was done with these components, except that partitioned loss was added

---

1The reason for this is that these were earlier experiments, done before switching to I3D and Transformer, and these experiments have not be redone with the new architecture.
<table>
<thead>
<tr>
<th>Multi-clip</th>
<th>Multi-clip + temporal encodings</th>
<th>Joint training</th>
</tr>
</thead>
<tbody>
<tr>
<td>Using 20% of data</td>
<td>21.7%</td>
<td>24.5%</td>
</tr>
<tr>
<td>Using 100% of data</td>
<td>42.8%</td>
<td>53.5%</td>
</tr>
</tbody>
</table>

Table 1: Testing the main approach.

<table>
<thead>
<tr>
<th>Multi-clip + temporal encodings</th>
<th>Joint training</th>
<th>Joint training with slow start</th>
</tr>
</thead>
<tbody>
<tr>
<td>5 clips used for evaluation</td>
<td>91.2%</td>
<td>73.8%</td>
</tr>
<tr>
<td>40 clips used for evaluation</td>
<td>92.7%</td>
<td>—</td>
</tr>
</tbody>
</table>

Table 2: Testing approach on pretrained weights.

<table>
<thead>
<tr>
<th>Multi-clip + temporal encodings reference</th>
<th>Fully-supervised reference</th>
<th>Semi-supervised</th>
</tr>
</thead>
<tbody>
<tr>
<td>25.7%</td>
<td>26.3%*</td>
<td>24.6%</td>
</tr>
</tbody>
</table>

Table 3: Testing semi-supervised learning.

<table>
<thead>
<tr>
<th>Baseline</th>
<th>Multi-clip</th>
<th>Multi-clip + temporal encodings</th>
<th>Joint training</th>
<th>Partitioned loss</th>
<th>Partitioned loss with classification restricted</th>
</tr>
</thead>
<tbody>
<tr>
<td>31.2%</td>
<td>37.9%</td>
<td>37.5%</td>
<td>37.3%</td>
<td>38.6%</td>
<td>39.3%</td>
</tr>
</tbody>
</table>

Table 4: Testing partitioned loss.

to the global representation and the experiments were done only for 100% of data. The use of the 3D convnet and LSTM instead of 13D and Transformer produces lower accuracy; the results of this section are not directly comparable to those in Section 3.1. To apply the partitioned loss in these experiments, we partition the global representation (of size 512) into 6 parts: 5 parts are as shown in Figure 4 and we add an additional “free” partition that no contrastive or similarity loss is applied to. We do not enforce a similarity constraint between videos of different classes, since they are ostensibly unrelated, other than that they are from the same dataset. We use similarity and contrastive losses based on cosine similarity. If $v_1, v_2$ are global representations for two videos, the similarity loss is simply $L_{\text{sim}}(v_1, v_2) = 1 - \cos(v_1, v_2)$ and the contrastive loss is $L_{\text{contr}}(v_1, v_2) = \cos(v_1, v_2)$, where $\cos(v_1, v_2) = v_1 \cdot v_2 / (||v_1|| \cdot ||v_2||)$ denotes the cosine similarity between $v_1$ and $v_2$. In addition to training the partitioned loss jointly with classification and prediction, we also run an experiment where we restrict classification to only use the partitions of the global representation that are expected to be useful for action classification as well as the free part of the representation. The partitions that would be expected to have relevance to action classification are: (1) similarity between videos of the same class, and (2) differences between videos of different classes. The results of the series of experiments is shown in Table 4.

The fact that restricting classification to some of the partitions provides evidence that the partitioned loss is “factoring out” information that is not relevant to classification.

4 Conclusions and Future Work

We have presented a new approach to predictive coding which improves accuracy on action recognition in the fully-supervised case as well a new approach to utilizing data structure for representation learning. One simple future exploration would be to focus on the multi-clip + temporal encoding aspect and study how number of frames per clip and the number of clips taken affects the accuracy, possibly over different encoder, aggregator, and predictor head architectures. This especially applies to the predictor head—although a feedforward network is a universal approximator, it does not seem to have the right structure to effectively condition on the temporal encoding. An attention-based model would probably be better. Any improvements in the multi-clip approach would be expected to produce gains for joint training as well. Another, more significant, direction would be to determine what causes the accuracy decrease in the semi-supervised case and/or modify the way the unlabelled data is incorporated such that it can improve accuracy, since it seems very reasonable to expect that it is possible to successfully incorporate semi-supervised learning based on the increase observed by joint training. One possible direction for partitioned loss would be a more thorough testing of how different partitions affect the final classification accuracy. It might also be useful to look at partitioned loss in an unsupervised or semi-supervised setting, perhaps on different types of datasets as well. Finally, self-supervised learning of a hierarchical structure on a single large example of sequential data via progressively larger scales of predictive coding (for example formatting a movie into scenes, cuts, and shots). More specifically: coding can be done for short time-scales over the entire example; then the coding loss can be computed at every timestep, splitting the data at points where the coding loss is higher; this can be continued for progressively larger time-scales of predictive coding (which would be expected to create progressively large segments of data) until predictive coding cannot successfully learn to make predictions. (note: it may be necessary to adjust the type of loss used as the scale increases). If such a structure could be used in a self-supervised manner and partitioned loss is effective in unsupervised settings, then
the two ideas could be combined to learn representations from long examples of sequential data without needing any further information about the data at all (the main significance of this would be that it would make it possible to learn structure and representations from easily-recorded data that is typical of normal human sensory input, e.g. the video or audio taken from person’s progression throughout their day).

References


(a) Classification based on single clip.

(b) Multi-clip approach without temporal encodings
(c) Multi-clip approach with temporal encodings.

(d) Joint training with partitioned loss.

Figure 6: Architectures corresponding to sequence of experiments in Section 3.1.