PARAMETER EFFICIENT MULTIMODAL TRANSFORMERS FOR VIDEO REPRESENTATION LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

Transformers have delivered impressive results in various language understanding tasks. This has motivated adapting pretrained language Transformers to a multimodal setting, where a new visual model is trained in tandem with the language model. However, existing work typically fixes the pretrained language model and train only the vision module, which limits its ability to learn cross-modal information. This is partly because of excessive memory requirements from over-parameterized multimodal Transformers. In this work, we focus on reducing the parameters of multimodal Transformers in the context of audio-visual video representation learning. We alleviate the high memory requirement by sharing the weights of Transformers across layers and modalities; we decompose the Transformer into modality-specific and modality-shared parts so that the model learns the dynamics of each modality both individually and together, and propose a novel parameter sharing scheme based on low-rank approximation. We show that our approach reduces parameters up to 80$\%$, allowing us to train our model end-to-end from scratch. To demonstrate our approach, we pretrain our model on 30-second clips from Kinetics-700 and transfer it to audio-visual classification tasks.

1 INTRODUCTION

Learning multimodal representation from unlabeled videos has received considerable attention (Baltrušaitis et al., 2018). Audio-visual learning is of particular interest due to the abundance of videos with natural audio-visual co-occurrence (Owens & Efros, 2018; Owens et al., 2018; Arandjelovic & Zisserman, 2018; Ephrat et al., 2018; Gao & Grauman, 2019; Alwassel et al., 2019). However, existing approaches learn localized representations from short videos (hundreds of milliseconds to just under a few seconds), capturing only *short-term* dependencies in data. While this is useful for certain applications, e.g., source separation (Ephrat et al., 2018) and atomic action recognition (Gu et al., 2018), learning representation that captures *long-term* dependencies is equally important, e.g., for activity recognition (Kay et al., 2017; Carreira et al., 2019; Sigurdsson et al., 2016). Unfortunately, processing long videos requires large memory resource and capturing long-term dependencies is a long-standing problem (Hochreiter & Schmidhuber, 1997; Cho et al., 2014; Vaswani et al., 2017).

In language understanding, strong progress has been made in large-scale learning of contextualized language representations using Transformers (Vaswani et al., 2017; Howard & Ruder, 2018; Peters et al., 2018; Radford et al., 2018; 2019; Devlin et al., 2019; Liu et al., 2019; Yang et al., 2019). Riding on the success of Transformers, several recent works have extended it to the multimodal setting by adding an additional vision module to the Transformer framework (Sun et al., 2019b; Lu et al., 2019). However, these models are typically not end-to-end trained; they rely on a language-pretrained BERT (Devlin et al., 2019), which is fixed throughout, and train only the visual components. While the pretrained BERT helps accelerate convergence and brings reliable extra supervision signal to the vision component, this partial learning setup can be undesirable if the text data comes from different distributions (of topics, dialects, or foreign languages) or if we want to apply it to different modalities (e.g., audio-visual). Unfortunately, end-to-end training of such multimodal Transformer architectures is challenging for most existing compute environments due to the excessive memory requirement.

In this work, we make three key contributions. First, we propose an *end-to-end trainable* bidirectional transformer architecture that learns contextualized audio-visual representations of long videos. Our model, shown in Figure 1, consists of audio/visual CNNs, audio/visual Transformers, and a multi-

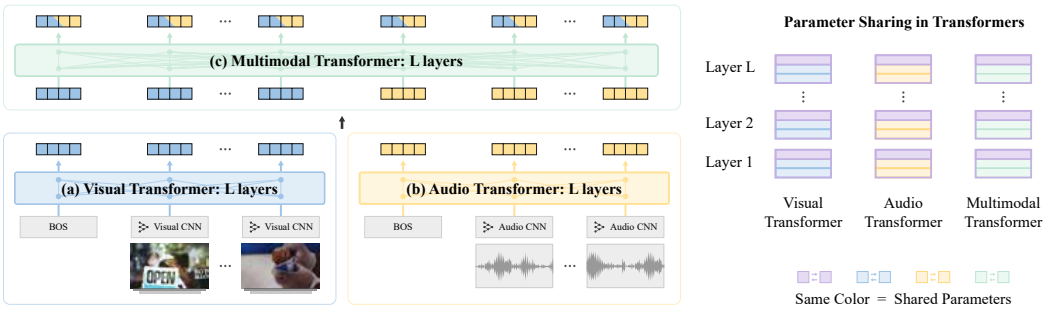


Figure 1: **(Left)** Our model consists of CNNs encoding short-term dynamics of each modality and Transformers encoding long-term dynamics of audio-visual information from videos. **(Right)** To alleviate excessive memory requirements, we propose an efficient parameter sharing scheme based on matrix decomposition with low-rank approximation, which allows us to train our model end-to-end.

modal Transformer. The CNNs operate on short (e.g., one second) video clips and are intended to capture short-term dynamics within each modality. The Transformer layers operate on long video sequences (e.g., 30 seconds), capturing long-term dynamics. To enable end-to-end training, we propose a novel parameter reduction technique that shares parts of weight parameters across Transformers and across layers within each Transformer. We show that this results in up to 80% parameter reduction, enabling end-to-end training of our model, with a minimal performance degradation. To the best of our knowledge, our work is the first to report end-to-end trained multimodal Transformers, and the first to apply Transformers for audio-visual representation learning.

The quality of negative samples is crucial in contrastive learning, which is part of our learning objective. As our second contribution, we propose a *content-aware* negative sampling strategy that favors negatives sufficiently similar to a positive instance. Our approach measures the similarity by reusing the CNN embeddings obtained during model training, and thus do not introduce extra parameters to learn. We show that this improves performance over the standard sampling strategies.

Our third contribution is a systematic evaluation of different modality fusion strategies. Existing works on multimodal BERT (all using vision-and-language data) typically apply one fusion strategy without thoroughly comparing with alternatives, e.g., some works perform early fusion (Sun et al., 2019b; Su et al., 2020) while others perform mid-level fusion (Lu et al., 2019; Tan & Bansal, 2019). As a result, it is unclear how different fusion methods affect the final performance. In this work, we compare three fusion strategies (early, mid, late) and show the superiority of mid-level fusion.

To demonstrate our approach, we pretrain our model on long (30-second) video clips from Kinetics-700 (Carreira et al., 2019) and finetune it on various video classification tasks. One benefit of the modular design of our architecture is flexibility: once pretrained, we can use any of the subnetworks for downstream tasks depending on the modalities involved (audio-only, visual-only, audio-visual) and video lengths (short and long). To show this, we evaluate our model on UCF101 (Soomro et al., 2012) and ESC50 (Gemmeke et al., 2017) for short-term visual/audio classification, and Charades (Sigurdsson et al., 2016) and Kinetics-Sounds (Arandjelovic & Zisserman, 2017) for long-term audio-visual action recognition.

2 Approach

Figure 1 shows an overview of the proposed model architecture. The input to our model is a sequence of visual clips $\mathbf{v}_{1:T}$ and the corresponding sequence of audio streams $\mathbf{a}_{1:T}$. For example, each sequence is a 30 second-long video divided into 30 non-overlapping clips (each clip is one second long). We divide our model into three parts with different characteristics, which are explained below.

Local Feature Embedding. We feed each of T video clips to a visual CNN $f_V(\mathbf{v}_t)$ to obtain $\mathbf{x}^v_{1:T} \in \mathbb{R}^{T\times D}$, and each audio stream to an audio CNN $f_A(\mathbf{a}_t)$ to obtain $\mathbf{x}^a_{1:T} \in \mathbb{R}^{T\times D}$.[1] Intuitively, the CNN outputs are *temporally local* embeddings as they have access to only a short-range temporal window of the entire video sequence. Thus, they are suitable for representing short-range atomic

[1]For notational simplicity, we drop the subscripts to refer to the entire sequence unless distinction is necessary.

actions (e.g., sit down, raise arms) that constitute long-range events (e.g., gym workout). We use the SlowFast network (Feichtenhofer et al., 2019) with a ResNet-50 backbone (He et al., 2016) as a visual CNN f_V, and a ResNet-50 as an audio CNN f_A. The weights of both CNNs are randomly initialized and trained end-to-end with the Transformer layers.

Unimodal Contextualized Embedding. The local feature embeddings capture short-term dynamics but lack long-term contextual information. We use Transformers (Vaswani et al., 2017) to enrich the embeddings with sequence-level context. We start by learning *unimodal* contextualized representations using the visual Transformer g_V and the audio Transformer g_A, respectively.

The Transformer consists of L layers, each with two sub-layers: a multi-head attention layer and a feed-forward layer. Given an input sequence of embeddings $\mathbf{x} \in \mathbb{R}^{T\times D}$ and A attention heads, the j-th head in the attention layer computes the output embedding sequence $\mathbf{a}_j \in \mathbb{R}^{T\times\gamma}, \gamma = D/A$ as

$$\mathbf{a}_j = \text{softmax}\left(\frac{QK}{\sqrt{\gamma}}\right)V, \qquad Q = \mathbf{x}W_j^q, K = \mathbf{x}W_j^k, V = \mathbf{x}W_j^v \tag{1}$$

where $W_j^q, W_j^k, W_j^v \in \mathbb{R}^{D\times\gamma}$ are weight matrices for computing the (`query`, `key`, `value`) triplet given the input $\mathbf{x}$. This operation is repeated for each attention head, and the outputs are combined (with concatenation followed by one linear layer with weights $W^b \in \mathbb{R}^{D\times D}$), producing $\mathbf{a} \in \mathbb{R}^{T\times D}$. Next, the feed-forward layer takes this intermediate output and computes $\mathbf{o} \in \mathbb{R}^{T\times D}$ using a two-layer fully-connected network with weights $W^c \in \mathbb{R}^{D\times E}$ and $W^d \in \mathbb{R}^{E\times D}$. The output of each sub-layer is computed using a residual function followed by layer normalization (Ba et al., 2016), i.e., $\text{LayerNorm}(x + \text{Sublayer}(x))$. In this work, we set the number of layers $L = 6$, the number of attention heads $A = 12$, the feature dimension $D = 768$ and the intermediate dimension $E = 3072$. For simplicity, we use this design for all layers across all three Transformers in our model.

Before feeding local embeddings $\mathbf{x}^v$ and $\mathbf{x}^a$ to unimodal Transformers, we augment them with "positional" embeddings. Specifically, we append to the beginning of each sequence a special vector `BOS` (beginning of sequence), i.e., $\mathbf{x}_0^v$ for visual and $\mathbf{x}_0^a$ for audio streams; their dimensions are same as $\mathbf{x}_t^v$ and $\mathbf{x}_t^a$, respectively. We also define positional embeddings $\mathbf{p}_{0:T}$ encoding time indices (we call this "time" embedding). This is necessary to preserve information about temporal ordering of local feature embeddings, which is otherwise lost in Eqn. 1. We combine them via layer normalization,

$$\mathbf{u}_t^v = \text{LayerNorm}(\mathbf{x}_t^v + \mathbf{p}_t^v), \quad \mathbf{u}_t^a = \text{LayerNorm}(\mathbf{x}_t^a + \mathbf{p}_t^a), \quad \forall t \in [0, T] \tag{2}$$

We initialize $\{\mathbf{x}_0^v, \mathbf{x}_0^a, \mathbf{p}_{0:T}^v, \mathbf{p}_{0:T}^a\}$ to the normal distribution and train them with the rest of the model.

We feed the augmented visual embeddings into the visual Transformer g_V and obtain $\mathbf{y}_{0:T}^v = g_V(\mathbf{u}_{0:T}^v)$, and similarly obtain $\mathbf{y}_{0:T}^a = g_A(\mathbf{u}_{0:T}^a)$. The embeddings at each time step has a direct access to the entire input sequence regardless of their position (it has a one-step signal path during forward and backward inference). Multiple layers of such feature transformation thus allow the resulting embedding to be deeply contextualized in the time dimension. We denote the output embeddings corresponding to the `BOS` positions by $\texttt{BOS}_g^v = \mathbf{y}_0^v$ and $\texttt{BOS}_g^a = \mathbf{y}_0^a$, and designate them as the *summary* embeddings representing the sequence of each modality.

Multimodal Contextualized Embedding. The unimodal embeddings capture long-term temporal context but miss out on cross-modal information. The final step in forward inference is to use a multimodal Transformer h_{AV} to obtain embeddings contextualized in the audio-visual space.

We first augment the embeddings $\mathbf{y}_{0:T}^v$ and $\mathbf{y}_{0:T}^a$ with modality and time embeddings. The modality embeddings $\mathbf{m}^v$ and $\mathbf{m}^a$ are vectors of the same dimension as $\mathbf{y}_t^v$ and $\mathbf{y}_t^a$, respectively. We share $\mathbf{m}^v$ (and $\mathbf{m}^a$) across all the unimodal embeddings $\mathbf{y}_{0:T}^v$ (and $\mathbf{y}_{0:T}^a$); thus, they add modality-discriminative information to the Transformer. We also add time embeddings $\mathbf{p}_{0:T}$ as before; however, unlike in the previous step, we share the same $\mathbf{p}_{0:T}$ between embeddings from the two modalities to correctly indicate the time indices. We augment the modality and time embeddings via layer normalization,

$$\mathbf{w}_t^v = \text{LayerNorm}(\mathbf{y}_t^v + \mathbf{p}_t + \mathbf{m}^v),\ \mathbf{w}_t^a = \text{LayerNorm}(\mathbf{y}_t^a + \mathbf{p}_t + \mathbf{m}^a),\ \forall t \in [0, T] \tag{3}$$

We feed the augmented visual embeddings $\mathbf{w}_{0:T}^v$ and audio embeddings $\mathbf{w}_{0:T}^a$ to the multimodal Transformer h_{AV}, one after another, and obtain $\mathbf{z}_{0:(2T+1)} = h_{AV}([\mathbf{w}_{0:T}^v; \mathbf{w}_{0:T}^a])$. We again denote the output embeddings corresponding to the `BOS` positions by $\texttt{BOS}_h^v = \mathbf{z}_0^v (= \mathbf{z}_0)$ and $\texttt{BOS}_h^a = \mathbf{z}_0^a (= \mathbf{z}_{T+1})$, and use them as summary embeddings encoding multimodal context.

We emphasize the importance of feeding $\mathbf{w}^v_{0:T}$ and $\mathbf{w}^a_{0:T}$ one after another. An alternative would be concatenating them before feeding them to h_{AV} and obtaining an output $\mathbf{z}_{0:T}$ (instead of $\mathbf{z}_{0:(2T+1)}$). However, this restricts the Transformer to access audio-visual embeddings only from the same time slices, which could be problematic when there is a temporally asynchronous relationship between the two modalities (e.g., a visual clip matches with sound captured a few times steps before). By arranging the two sequences one after the other, the Transformer can mix-and-match appropriate audio-visual embeddings in an asynchronous manner. Another practical concern with the alternative approach is that it significantly increases the model size; the weight matrices W_q, W_k, W_v grow quadratically with the input feature dimension D. Serializing the input resolves both issues.

2.1 Self-Supervised Pretraining Objectives

Task 1: Masked Embedding Prediction (MEP). BERT (Devlin et al., 2019) is trained using the masked language model (MLM) task, which randomly selects input tokens and replaces them with a mask token. The model is then trained to predict the original (unmasked) tokens by solving a classification task with a cross-entropy loss. However, inputs to our model are real-valued audio-visual signals (rather than discrete tokens), so applying the MLM task requires input discretization, which causes information loss (Lu et al., 2019; Sun et al., 2019a). We instead train our model to identify the correct visual clip or audio stream compared to a set of negative samples in a contrastive manner, which does not require input discretization.

We formulate our MEP task using InfoNCE (Oord et al., 2018), which is the softmax version of the noise contrastive estimation (NCE) (Gutmann & Hyvärinen, 2010). Let $\tilde{\mathbf{o}}_t$ be the t-th output of any of the three Transformers obtained by masking the t-th input $\mathbf{x}_t$. Our InfoNCE loss is then defined as

$$\mathcal{L}_{\text{NCE}}(\mathbf{x}, \tilde{\mathbf{o}}) = -\mathbb{E}_{\mathbf{x}} \left[\sum_t \log \frac{\mathrm{I}(\mathbf{x}_t, \tilde{\mathbf{o}}_t)}{\mathrm{I}(\mathbf{x}_t, \tilde{\mathbf{o}}_t) + \sum_{j \in \text{neg}(t)} \mathrm{I}(\mathbf{x}_j, \tilde{\mathbf{o}}_t)} \right], \tag{4}$$

where $\text{neg}(t)$ are negative sample indices and the compatibility function $\mathrm{I}(\mathbf{x}_t, \tilde{\mathbf{o}}_t)$ is,

$$\mathrm{I}(\mathbf{x}_t, \tilde{\mathbf{o}}_t) = \exp\left(\text{FFN}(\tilde{\mathbf{o}}_t) W_I \mathbf{x}_t\right), \tag{5}$$

where $W_I \in \mathbb{R}^{P \times D}$ ($P = 256$) and FFN is a two-layer feed-forward network. The use of a non-linear prediction head has shown to improve the quality of the representations learned in a contrastive learning setup (Chen et al., 2020); following the recent work in Transformers (Devlin et al., 2019; Liu et al., 2019; Lan et al., 2020), we use a GELU non-linear activation function (Hendrycks & Gimpel, 2016) in FFN. Optimizing Eqn. 4 enforces $\mathrm{I}(\mathbf{x}_t, \tilde{\mathbf{o}}_t)$ to approximate the density ratio $\frac{p(\mathbf{x}_t|\tilde{\mathbf{o}}_t)}{p(\mathbf{x}_t)}$; this can be seen as maximizing the mutual information between $\mathbf{x}_t$ and $\tilde{\mathbf{o}}_t$ (Oord et al., 2018). Intuitively, this encourages the Transformer to capture the underlying dynamics of $\mathbf{x}$ from each modality without explicitly learning a generative model $p(\mathbf{x}_t|\tilde{\mathbf{o}}_t)$.

Negative sampling. We find that a good negative sampling strategy is essential for the model's convergence. Existing approaches either use all but $\mathbf{x}_t$ (positive) within a mini-batch as negative samples or limit it to the current sequence only. However, both these methods ignore the data content and thus can miss useful negatives. Oord et al. (2018) showed that leveraging prior knowledge about data can improve the negative sample quality (e.g., by sampling negatives from the same speaker as the positive). Unfortunately, such prior knowledge is often not available in unlabeled videos.

We propose a *content-aware* negative sampling strategy that favors negatives sufficiently similar to a positive instance in the CNN embedding space. We compute a pairwise ℓ_2 distance between $\mathbf{x}_t$ (positive) and all other instances within a mini-batch, and normalize them to the [0, 1] interval. We discard instances that fall outside the 95% confidence interval in the normalized distance space, and sample the negatives from the remainder using the normalized distance as sampling probability. This makes instances similar to the positive instance have more chance to become negatives. Our approach is inspired by Ulyanov et al. (2018), who showed randomly initialized CNNs provide a strong prior over natural images. This suggests that our local feature embeddings $\mathbf{x}^v$ (and $\mathbf{x}^a$) capture inherent regularities in video clips (and audio streams), and thus the distance measured on them can approximate content dissimilarity (this will improve as the training progresses).

Finally, our MEP loss is the InfoNCE loss computed on all three Transformers,

$$\mathcal{L}_{\text{MEP}} = \mathcal{L}_{\text{NCE}}(\mathbf{x}^a, \tilde{\mathbf{y}}^a) + \mathcal{L}_{\text{NCE}}(\mathbf{x}^v, \tilde{\mathbf{y}}^v) + \mathcal{L}_{\text{NCE}}([\mathbf{x}^a; \mathbf{x}^v], \tilde{\mathbf{z}}) \tag{6}$$

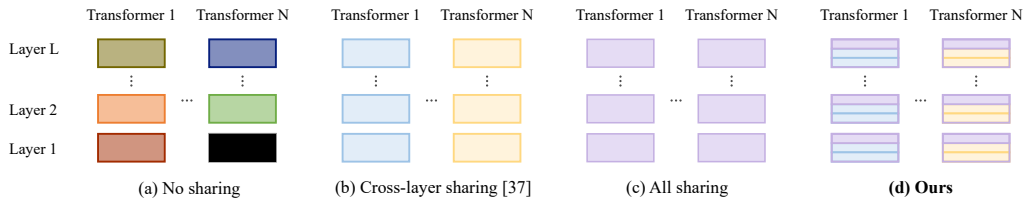


Figure 2: Comparison of parameter sharing schemes. Ours combines (b) and (c) but decomposes weights in each layer into private and shared parts so only the latter is shared across Transformers.

Task 2: Correct Pair Prediction (CPP). The MEP task encourages our model to learn the underlying dynamics within each modality. To help our model learn cross-modal dynamics, we design a task that predicts whether a pair of audio-visual embeddings is from the same video. Specifically, we define two binary classifiers, one for the two unimodal Transformers and another for the multimodal Transformer. Each classifier takes as input either $\mathbf{s}_g = [\mathbf{y}_0^v; \mathbf{y}_0^a]$ (and $[\mathbf{z}_0^v; \mathbf{z}_0^a]$), a pair of audio-visual "summary" embeddings corresponding to the BOS positions, or $\mathbf{s}_h = [\mathbf{y}_t^v; \mathbf{y}_t^a]$ (or $[\mathbf{z}_t^v; \mathbf{z}_t^a]$), the output embeddings sampled at random positions (we take two random positions $t \in [1, T]$). The classifier predicts $p(c|\mathbf{s})$ indicating whether the pair is from the same video ($c = 1$) or from different videos ($c = 0$). We train the classifiers with a binary cross-entropy loss,

$$\mathcal{L}_{CPP} = -\mathbb{E}_{\mathbf{x},\mathbf{y}} \left[c \cdot \log p(c|\mathbf{s}_g) + c \cdot \log p(c|\mathbf{s}_h) \right] \tag{7}$$

where $\cdot$ is the inner product. We generate a random derangement of the input mini-batch so that the number of positive and negative pairs are guaranteed to be the same.

Overall Pretraining Objective. We train our model end-to-end from scratch by optimizing $\mathcal{L}_{MEP} + \alpha\mathcal{L}_{CPP}$ with a balancing term α. We find our model is insensitive to this term, so we set $\alpha = 1.0$.

2.2 Parameter Reduction

Optimizing our model is challenging due to the large memory requirement. The most expensive part is the Transformers, which take up 76% of model parameters. One could reduce the model size by making the Transformers shallower, but the depth of Transformers has shown to be crucial to get good performance (Devlin et al., 2019). We propose to reduce the model size by aggressively sharing parts of weights across Transformers as well as layers within each Transformer (see Figure 2 (d)).

Sharing across Transformers. We first consider sharing weights across Transformers. Each Transformer encodes data coming from different distributions: g_V encodes $\mathbf{x}^v$, g_A encodes $\mathbf{x}^a$, and h_{AV} encodes $(\mathbf{y}^v, \mathbf{y}^a)$. These input distributions may each exhibit different dynamics, yet together share certain regularities because they all come from the same videos. Motivated by this, we decompose Transformer weights into shared and private parts so that different patterns can be learned in a parameter-efficient manner. Recall that each layer of a Transformer contains weights $\{W^q, W^k, W^v, W^b, W^c, W^d\}$. We decompose each of these weights into $W = U\Sigma V$, where $W \in \mathbb{R}^{M\times N}, U \in \mathbb{R}^{M\times O}, \Sigma \in \mathbb{R}^{O\times O}, V \in \mathbb{R}^{N\times O}$. We perform low-rank approximation of W by setting the rank $O \ll M, N$, and share U across Transformers while keeping Σ and V private to each Transformer. This helps reduce parameters because $MO + 3(O^2 + NO) \ll 3MN$. We experimented with different matrix ranks O but the differences were small; we set $O = 128$ ($M, N = 768$ or 3072).

The decomposition converts a linear projection of input $W\mathbf{x}$ into a series of (unconstrained) linear projections $U\Sigma V\mathbf{x}$. However, this can cause numerical instability during optimization (Nocedal & Wright, 2006). We could perform the Singular Value Decomposition (SVD) over W so that it performs rotation (V), stretch (Σ), and rotation (U) with orthogonal vectors in U and V. Unfortunately, solving the full SVD has a computational complexity of $\mathcal{O}(\max(M, N)^2)$ (Golub & Van Loan, 2012). Here, we put an orthogonality constraint only on Σ and perform projection (V), rotation (Σ), and projection (U) of input $\mathbf{x}$. In addition, we put $V\mathbf{x}$ in a unit sphere (via ℓ_2-normalization) before rotating it with Σ. This not only improves numerical stability, but also removes magnitude information in $V\mathbf{x}$ and keeps angular information only, which has been shown to provide sample discriminative information (Chen et al., 2019a). To impose the orthogonality constraint, we use the Padé approximation with a scale-squaring trick of Lezcano-Casado and Martínez-Rubio (Lezcano-Casado & Martínez-Rubio, 2019). Intuitively, we project and then rotate the input ($\Sigma V\mathbf{x}$) in each Transformer so that it captures the

Fusion Method	Audio-Visual	Audio-only	Visual-only
Early	64.9 / 89.8	- / -	- / -
Late-w/-CPP	61.0 / 88.7	52.3 / 80.8	41.0 / 71.3
Late-w/o-CPP	60.6 / 87.6	50.5 / 79.9	40.7 / 71.7
Mid†	**65.7 / 89.9**	**53.5 / 82.7**	**42.5 / 73.2**

Sampling Method	top-1	top-5
Current-Sequence	64.6	89.8
Current-MiniBatch	65.5	90.8
CANS-Dissimilar	66.2	91.1
CANS-Similar†	**67.5**	**92.3**

Model	X.-L	X.-T	Params	top-1/5
Multi-6	✗	✗	155M	- / -
Multi-6	✓	✗	49M	65.7 / 89.9
Multi-6	✓	✓(All)	34M	67.1 / 92.3
Multi-6	✓	✓(Part†)	**31M**	**67.5 / 92.3**

Model	X.-L	X.-T	Params	top-1/5
Vis-2	✗	✗	38M	41.4 / 71.0
Vis-2	✓	✗	31M	41.2 / 72.9
Vis-6	✗	✗	67M	43.8 / 74.2
Vis-6	✓	✗	31M	43.5 / 73.7

Table 1: Ablation study on Kinetics-Sounds (Arandjelovic & Zisserman, 2017) comparing: **(a; top-left)** multimodal fusion methods, **(b; top-right)** negative sampling strategies, and **(c; bottom)** parameter sharing schemes. X.-L: Cross-layer, X.-T: Cross-Transformer sharing. We report top-1 and top-5 accuracy (%). †: Ours.

dynamics of each input distribution independently. We then project it to the shared space via U, capturing shared regularities across all three Transformers.

Sharing across Layers. Recently, Bai et al. (2019a) showed that sharing parameters across layers in deep neural networks does not hurt the representational power of the network. Furthermore, (Lan et al., 2020) demonstrated that cross-layer parameter sharing in the Transformer leads to a lighter and faster-to-train model without sacrificing the performance on various language understanding benchmarks. Motivated by this, we let each Transformer share parameters across different layers.

3 Experiments

We pretrain our model on Kinetics-700 (Carreira et al., 2019) or AudioSet (Gemmeke et al., 2017) and finetune it on various downstream tasks. The official release of Kinetics-700 contains 10-second clips only, so we download 410K original videos from YouTube and take 30-second clips from each video. For fair comparison with prior work, we use 10-second clips from the official release of AudioSet (we used 1.8M clips). We pretrain our model on 64 NVIDIA Tesla V100 GPUs with a batch size of 256 for 220K iterations. For downstream tasks, we evaluate on *short*-video/audio classification using UCF-101 (Soomro et al., 2012) (13K clips from 101 classes; 7.2 seconds on average) and ESC-50 (Gemmeke et al., 2017) (2K clips from 50 classes; 5 seconds), and on *long*-video classification using Kinetics-Sounds (Arandjelovic & Zisserman, 2017) (23K videos from 32 classes; 10 seconds on average) and Charades (Sigurdsson et al., 2016) (10K videos from 157 classes; 30 seconds on average). We describe various details about experimental setup in Appendix. Code will be released.

3.1 Results and Discussion

Multimodal Fusion Methods. To evaluate different fusion methods on the quality of learned representation, we test the following settings: (i) Early uses a single multimodal Transformer with $2 \times L$ layers, (ii) Mid is our approach described in Figure 1, (iii) Late uses two unimodal Transformers each with $2 \times L$ layers. All the methods are pretrained on audio-visual data using CPP and MEP losses, except for (iv) Late-w/o-CPP where we use only the MEP loss. We finetune the pretrained models on audio-visual, audio-only, and visual-only scenarios. For fair comparisons across different fusion methods, we do not perform parameter sharing in this ablation setting.

Table 1 (a) shows that Early and Mid outperform Late on the audio-visual scenario. This suggests the importance of encoding cross-modal information. Note that Late-w/-CPP gets cross-modal self-supervision, which gives marginal performance improvement over Late-w/o-CPP; however, both methods miss the opportunity to *encode* any cross-modal relationship, leading to inferior results. While both Early and Late perform similarly in the audio-visual scenario, only Late can be used in unimodal downstream scenarios (c.f., Early requires the presence of both modalities). This has practical implications: Mid and Late can effectively handle missing modalities, i.e., once pretrained on audio-visual data, we can use it on any of audio-visual, audio-only, and visual-only scenarios. Our Mid fusion approach enjoys both the advantages, i.e., learning cross-modal relationship and being robust to missing modalities, achieving overall the best performance.

Negative Sampling Strategies. We compare four strategies: (i) `Current-Sequence` takes all but the positive instance from the same sequence as negatives, (ii) `Current-MiniBatch` takes all but the positive instance in the mini-batch as negatives; this subsumes `Current-Sequence`, (iii) `CANS-Dissimilar` stochastically samples negatives using a modified version of our content-aware negative sampling (CANS) that favors *dissimilar* samples, and (iv) `CANS-Similar` is our proposed CANS approach that favors negatives that are *similar* to the positive instance.

Table 1 (b) shows `Current-Sequence` is the least effective: It makes MEP too difficult because negatives are (sometimes too much) similar to positives. As a result, the training dynamics is dominated by CPP, which is relatively easier, leading to inferior performance. We make quite the contrary observations from `Current-MiniBatch`: the inclusion of negatives from different videos makes MEP easier and thus makes it dominate the training dynamics. Our `CANS` approach solves both these issues by eliminating negatives that are either almost identical to or trivial to distinguish from the positives, based on the 95% CI over the CNN embedding distances. It also samples negatives in a stochastic manner so a wide variety of samples can be included as negatives. Our proposed `CANS-Similar` can be considered as a "softened" version of `Current-Sequence`; it samples negatives that are similar to positives with a high probability (this can be considered as online hard negative mining), but it also takes instances from different videos with a lower probability. This balances out hard and easy negatives, making the MEP task effective.

Parameter Sharing Schemes. Our parameter reduction scheme reduces the model size from 155M to 31M (by 80%) (Table 1 (a)). We reduce the model size by sharing weights across Transformers and across layers. We validate these ideas in two sets of experiments. Table 1 (c-left) compares cross-Transformer weight sharing schemes. We use `Multi-6` that uses all three Transformers with 6 layers each, and compare four methods that correspond to Figure 2 (a)-(d). Note that `No sharing` is too large to fit in a Tesla V100 GPU (16GB) even with 2 samples so we only report the number of parameters. We see that our proposed approach, `Part`, achieves the best performance with the least number of parameters. One might ask how `Part` leads to a smaller model when `All` shares all the weights across Transformers: We decompose weights $W = U\Sigma V$ with low-rank approximation and share only U across Transformers, while the ΣV part learns modality-specific dynamics. Table 1 (c-right) compares cross-layer weight sharing schemes using the visual Transformer with either 2 (`Vis-2`) or 6 (`Vis-6`) layers. The results show that sharing weights across layers does not hurt the performance, confirming the observations by Lan et al. (2020) in the audio-visual setting.

Pretraining Objectives. To evaluate the importance of MEP and CPP tasks, we test two settings: (i) `Mid-w/o-CPP` and (ii) `Mid-w/o-MEP`. On Kinetics-Sounds, these achieve 65.9% and 64.6%, respectively; ours achieve 67.5% (top-1 accuracy). The result show that the MEP task plays an important role during pretraining, confirming the findings from Sun et al. (2019a) that the InfoNCE loss, as deployed in CBT, is effective in the cross-modal setting. The result also shows that augmenting MEP with CPP provides further performance improvement by learning cross-modal correspondence.

Downstream Evaluation. We pretrain our model with `Mid` fusion using MEP and CPP tasks (with `CANS-Similar`), and employ `Part` weight sharing. We use either Kinetics-700 or AudioSet for fair comparisons with prior work. Table 2 (a)/(b) shows *short*-video/audio classification results on UCF-101/ESC-50. We use only the visual/audio CNN (no Transformers) and add a linear classifier on top, and finetune the model end-to-end. Although our model is pretrained on long video clips with no direct supervision to the CNN layers (gradients must flow *through* Transformers), it outperforms most of the baselines (except for AVTS on UCF-101) that received direct supervision from short video clips. We note that, similar to ours, CBT (Su et al., 2020) is a multimodal Transformer pretrained on long video clips and thus is the most meaningful comparison to ours; ours outperform CBT on UCF-101 by 5.7%. For sound classification, our approach outperform all existing published results.

Table 2 (c) shows *long*-video classification results on Charades and Kinetics-Sounds (KS) when pretrained on Kinetics-700. We test `Visual-only (V)`, `Audio-only (A)`, and `Multimodal (M)` settings to verify the benefit of multimodal learning. Because there is no published self-supervised learning results on these datasets, we demonstrate long-term representations by comparing CNNs (`CNN`; short-term) to Transformers (`BERT`; long-term) on KS that contains 10-second clips. Since CNNs process 1-second clips, we feed 10 non-overlapping clips to CNNs and average the prediction output. In all settings, we add a 2-layer MLP with softmax classifier on top. The results show that Transformers outperform CNNs on Kinetics-Sounds, suggesting the superiority of long-term representations. We also see that combining audio-visual information performs the best. We notice

Method	Net	Data	UCF
ST-Puzzle	3D-R18	K400	65.8
ClipOrder	R(2+1)D	UCF	72.4
DPC	3D-R34	K400	75.7
CBT	S3D	K600	79.5
MultiSens	3D-R18	AS	82.1
AVTS	MC3-18	K400	85.8
AVTS	MC3-18	AS	**89.0**
Ours	SlowFast	K700	85.2
Ours	SlowFast	AS	86.1

Method	Net	Data	ESC
SVM	MLP	-	39.6
ConvAE	CNN-4		39.9
RF	MLP	-	44.3
ConvNet	CNN-4	-	64.5
SoundNet	CNN-8	FS	74.2
L^3-Net	CNN-8	FS	79.3
DMC	VGG-ish	FS	79.8
AVTS	VGG-M	AS	80.6
Ours	ResNet-50	AS	**82.8**

Method	Charades	KS
Random	5.9	- / -
ATF	18.3	- / -
ATF (OF)	22.4	- / -
V-CNN	-	45.8 / 73.3
A-CNN	-	49.4 / 76.9
M-CNN	-	59.4 / 83.6
V-BERT	26.0	49.5 / 78.9
A-BERT	27.4	58.9 / 85.7
M-BERT	**29.5**	**75.6 / 94.6**

Datasets. K: Kinetics, AS: AudioSet, FS: Flicker-SoundNet, KS: Kinetics-Sounds. **Baselines.** ST-Puzzle (Kim et al., 2019), ClipOrder (Xu et al., 2019), DPC (Han et al., 2019), CBT (Sun et al., 2019a), MultiSens (Owens & Efros, 2018), AVTS (Korbar et al., 2018), AE (Aytar et al., 2016), SVM (Piczak, 2015a), RF (Piczak, 2015a), ConvNet (Piczak, 2015b), SoundNet (Aytar et al., 2016), L^3-Net (Arandjelovic & Zisserman, 2017), DMC (Hu et al., 2019), ATF (Sigurdsson et al., 2017)

Table 2: **(a; left):** Short video classification results on UCF101 (mean accuracy (%)). **(b; center):** Short audio classification results on ESC-50 (mean accuracy (%)). **(c; right):** Long video classification results on Charades (mAP) and Kinetics-Sounds (KS; top-1/5 accuracy (%)).

that audio representations are generally stronger than visual representations; we believe that learning discriminative visual representations is generally more challenging, especially when the CNNs receive (self-)supervision signals *only indirectly* through Transformers. We believe that providing (self-)supervision directly to CNNs, e.g., by first pretraining CNNs on 3D rotation prediction (Jing et al., 2018) and then jointly training the whole model (as was done in CBT (Sun et al., 2019a)), could further improve performance. Incorporating contrastive learning (Chen et al., 2020) over the CNN embeddings and training the whole model end-to-end is another promising direction for future work.

4 Related Work

Multimodal BERT. Extending BERT (Devlin et al., 2019) to vision-and-language has been actively studied. Existing work typically adopt early fusion (Li et al., 2019; Alberti et al., 2019; Sun et al., 2019b; Li et al., 2020; Zhou et al., 2020; Su et al., 2020; Chen et al., 2019b; Zhu & Yang, 2020) or mid fusion (Tan & Bansal, 2019; Lu et al., 2019; Sun et al., 2019a; Luo et al., 2020) without thorough validation, and they train only visual components while relying on a language-pretrained BERT. Compared to all the previous work, our approach is the first to extend BERT to the audio-visual setting, and the first to demonstrate a fully end-to-end trained multimodal BERT. This is enabled by our novel parameter reduction technique, which is one of our main technical contributions.

Audio-Visual Learning. Early work in audio-visual learning focused on speech signals, improving audio-visual speech recognition than unimodal approaches (Ngiam et al., 2011; Srivastava & Salakhutdinov, 2012). Recent approaches leverage unlabeled videos from specific domains (Owens et al., 2016; Gao & Grauman, 2019; Zhao et al., 2018; Ephrat et al., 2018) and often demonstrate on audio-visual source separation, localization, and co-segmentation. However, these approaches rely on short-term audio-visual correspondence and thus may not generalize to long-term video recognition that requires global context (as was suggested in (Hjelm et al., 2019)), which this work focuses on.

Parameter Reduction. Network pruning (Reed, 1993; Caron et al., 2020) trains a large model and then reduces its size while maintaining performance. Reducing the size of CNNs for mobile applications is an active research area (Rastegari et al., 2016; Howard et al., 2017; 2019; Zhang et al., 2018; Iandola et al., 2016). Our work is closely related to the work that shares parameters across layers in deep neural networks. Trellis network (Bai et al., 2019b) is a temporal convolutional architecture with weight-tying across time and depth. Similar to ours, Universal Transformer (Dehghani et al., 2019), RSNMT (Dabre & Fujita, 2019), DEQ (Bai et al., 2019a), ALBERT (Lan et al., 2020) share weights across layers in Transformers. We combine this idea with our novel cross-Transformer weight sharing, which decomposes weight matrices with low-rank approximation.

5 Conclusion

We introduced a multimodal bidirectional Transformer architecture for self-supervised learning of contextualized audio-visual representation from unlabeled videos. Our main technical contributions include: (1) we propose a parameter efficient multimodal Transformers based on matrix decomposition with low-rank approximation; (2) we propose a novel content-aware negative sampling technique for contrastive learning. We demonstrate a successful end-to-end training of multimodal Transformers for audio-visual learning (which is, to the best of our knowledge, the first time in the literature). We also report comprehensive evaluation of various design decisions in multimodal learning.

REFERENCES

Chris Alberti, Jeffrey Ling, Michael Collins, and David Reitter. Fusion of Detected Objects in Text for Visual Question Answering. In *EMNLP-IJCNLP*, 2019.

Humam Alwassel, Dhruv Mahajan, Lorenzo Torresani, Bernard Ghanem, and Du Tran. Self-Supervised Learning by Cross-Modal Audio-Video Clustering. *arXiv preprint arXiv:1911.12667*, 2019.

Relja Arandjelovic and Andrew Zisserman. Look, Listen and Learn. In *ICCV*, 2017.

Relja Arandjelovic and Andrew Zisserman. Objects that Sound. In *ECCV*, 2018.

Yusuf Aytar, Carl Vondrick, and Antonio Torralba. SoundNet: Learning Sound Representations from Unlabeled Video. In *NeurIPS*, 2016.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer Normalization. In *NeurIPS Deep Learning Symposium*, 2016.

Shaojie Bai, J. Zico Kolter, and Vladlen Koltun. Deep Equilibrium Models. In *NeurIPS*, 2019a.

Shaojie Bai, J Zico Kolter, and Vladlen Koltun. Trellis Networks for Sequence Modeling. In *ICLR*, 2019b.

Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. Multimodal Machine Learning: A Survey and Taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41(2): 423–443, 2018.

Mathilde Caron, Ari Morcos, Piotr Bojanowski, Julien Mairal, and Armand Joulin. Pruning Convolutional Neural Networks with Self-Supervision. *arXiv preprint arXiv:2001.03554*, 2020.

Joao Carreira, Eric Noland, Chloe Hillier, and Andrew Zisserman. A Short Note on the Kinetics-700 Human Action Dataset. *arXiv preprint arXiv:1907.06987*, 2019.

Beidi Chen, Weiyang Liu Animesh Garg, Zhiding Yu, Anshumali Shrivastava, Jan Kautz, and Anima Anandkumar. Angular Visual Hardness. In *ICML Workshop on Identifying and Understanding Deep Learning Phenomena*, 2019a.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A Simple Framework for Contrastive Learning of Visual Representations. *arXiv preprint arXiv:2002.05709*, 2020.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. UNITER: UNiversal Image-TExt Representation Learning. *arXiv preprint arXiv:1909.11740*, 2019b.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *EMNLP*, 2014.

Raj Dabre and Atsushi Fujita. Recurrent Stacking of Layers for Compact Neural Machine Translation Models. In *AAAI*, 2019.

Mostafa Dehghani, Stephan Gouws, Oriol Vinyals, Jakob Uszkoreit, and Łukasz Kaiser. Universal Transformers. In *ICLR*, 2019.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL-HLT*, 2019.

Ariel Ephrat, Inbar Mosseri, Oran Lang, Tali Dekel, Kevin Wilson, Avinatan Hassidim, William T Freeman, and Michael Rubinstein. Looking to Listen at the Cocktail Party: A Speaker-Independent Audio-Visual Model for Speech Separation. *ACM Transactions on Graphics*, 37(4), 2018.

Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. SlowFast Networks for Video Recognition. In *ICCV*, 2019.

Ruohan Gao and Kristen Grauman. 2.5D Visual Sound. In *CVPR*, 2019.

Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. Audio Set: An Ontology and Human-Labeled Dataset for Audio Events. In *ICASSP*, 2017.

Gene H Golub and Charles F Van Loan. *Matrix Computations*, volume 3. JHU press, 2012.

Chunhui Gu, Chen Sun, David A Ross, Carl Vondrick, Caroline Pantofaru, Yeqing Li, Sudheendra Vijayanarasimhan, George Toderici, Susanna Ricco, Rahul Sukthankar, et al. AVA: A Video Dataset of Spatio-temporally Localized Atomic Visual Actions. In *CVPR*, 2018.

Michael Gutmann and Aapo Hyvärinen. Noise-Contrastive Estimation: A New Estimation Principle for Unnormalized Statistical Models. In *AISTATS*, 2010.

Tengda Han, Weidi Xie, and Andrew Zisserman. Video Representation Learning by Dense Predictive Coding. In *ICCV Workshop on Large Scale Holistic Video Understanding*, 2019.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. In *CVPR*, 2016.

Dan Hendrycks and Kevin Gimpel. Gaussian Error Linear Units (GELUs). *arXiv preprint arXiv:1606.08415*, 2016.

R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. In *ICLR*, 2019.

Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural computation*, 9(8): 1735–1780, 1997.

Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for MobileNetV3. In *ICCV*, 2019.

Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications. *arXiv preprint arXiv:1704.04861*, 2017.

Jeremy Howard and Sebastian Ruder. Universal Language Model Fine-tuning for Text Classification. In *ACL*, 2018.

Di Hu, Feiping Nie, and Xuelong Li. Deep Multimodal Clustering for Unsupervised Audiovisual Learning. In *CVPR*, 2019.

Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size. *arXiv preprint arXiv:1602.07360*, 2016.

Longlong Jing, Xiaodong Yang, Jingen Liu, and Yingli Tian. Self-Supervised Spatiotemporal Feature Learning via Video Rotation Prediction. *arXiv preprint arXiv:1811.11387*, 2018.

Will Kay, Joao Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, et al. The Kinetics Human Action Video Dataset. *arXiv preprint arXiv:1705.06950*, 2017.

Dahun Kim, Donghyeon Cho, and In So Kweon. Self-Supervised Video Representation Learning with Space-Time Cubic Puzzles. In *AAAI*, 2019.

Bruno Korbar, Du Tran, and Lorenzo Torresani. Cooperative Learning of Audio and Video Models from Self-Supervised Synchronization. In *NeurIPS*, 2018.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. In *ICLR*, 2020.

Mario Lezcano-Casado and David Martínez-Rubio. Cheap Orthogonal Constraints in Neural Networks: A Simple Parametrization of the Orthogonal and Unitary Group. In *ICML*, 2019.

Gen Li, Nan Duan, Yuejian Fang, Min Gong, Daxin Jiang, and Ming Zhou. Unicoder-VL: A Universal Encoder for Vision and Language by Cross-modal Pre-training. In *AAAI*, 2020.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. VisualBERT: A Simple and Performant Baseline for Vision and Language. *arXiv preprint arXiv:1908.03557*, 2019.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A Robustly Optimized BERT Pretraining Approach. *arXiv preprint arXiv:1907.11692*, 2019.

Ilya Loshchilov and Frank Hutter. Decoupled Weight Decay Regularization. In *ICLR*, 2019.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks. In *NeurIPS*, 2019.

Huaishao Luo, Lei Ji, Botian Shi, Haoyang Huang, Nan Duan, Tianrui Li, Xilin Chen, and Ming Zhou. UniViLM: A Unified Video and Language Pre-Training Model for Multimodal Understanding and Generation. *arXiv preprint arXiv:2002.06353*, 2020.

Jiquan Ngiam, Aditya Khosla, Mingyu Kim, Juhan Nam, Honglak Lee, and Andrew Y Ng. Multimodal Deep Learning. In *ICML*, 2011.

Jorge Nocedal and Stephen Wright. *Numerical Optimization*. Springer Science & Business Media, 2006.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation Learning with Contrastive Predictive Coding. In *NeurIPS*, 2018.

Andrew Owens and Alexei A Efros. Audio-Visual Scene Analysis with Self-Supervised Multisensory Features. In *ECCV*, 2018.

Andrew Owens, Phillip Isola, Josh McDermott, Antonio Torralba, Edward H Adelson, and William T Freeman. Visually Indicated Sounds. In *CVPR*, 2016.

Andrew Owens, Jiajun Wu, Josh H McDermott, William T Freeman, and Antonio Torralba. Learning Sight from Sound: Ambient Sound Provides Supervision for Visual Learning. *International Journal of Computer Vision*, 126(10):1120–1137, 2018.

Daniel S Park, William Chan, Yu Zhang, Chung-Cheng Chiu, Barret Zoph, Ekin D Cubuk, and Quoc V Le. SpecAugment: A Simple Data Augmentation Method for Automatic Speech Recognition. In *INTERSPEECH*, 2019.

Mandela Patrick, Yuki M Asano, Ruth Fong, João F Henriques, Geoffrey Zweig, and Andrea Vedaldi. Multi-modal Self-Supervision from Generalized Data Transformations. *arXiv preprint arXiv:2003.04298*, 2020.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep Contextualized Word Representations. In *NAACL-HLT*, 2018.

Karol J. Piczak. ESC: Dataset for Environmental Sound Classification. In *ACM-MM*, 2015a.

Karol J. Piczak. Environmental Sound Classification with Convolutional Neural Networks. In *MLSP*, 2015b.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving Language Understanding by Generative Pre-Training. *Techincal Report, OpenAI*, 2018.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language Models are Unsupervised Multitask Learners. *OpenAI Blog*, 1(8), 2019.

Mohammad Rastegari, Vicente Ordonez, Joseph Redmon, and Ali Farhadi. XNOR-Net: ImageNet Classification Using Binary Convolutional Neural Networks. In *ECCV*, 2016.

Sashank J Reddi, Satyen Kale, and Sanjiv Kumar. On the Convergence of Adam and Beyond. In *ICLR*, 2018.

Russell Reed. Pruning Algorithms-A Survey. *IEEE Transactions on Neural Networks*, 4(5):740–747, 1993.

Gunnar A Sigurdsson, Gül Varol, Xiaolong Wang, Ali Farhadi, Ivan Laptev, and Abhinav Gupta. Hollywood in Homes: Crowdsourcing Data Collection for Activity Understanding. In *ECCV*, 2016.

Gunnar A Sigurdsson, Santosh Divvala, Ali Farhadi, and Abhinav Gupta. Asynchronous Temporal Fields for Action Recognition. In *CVPR*, 2017.

Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. UCF101: A Dataset of 101 Human Action Classes From Videos in The Wild. *CRCV-TR-12-01*, 2012.

Nitish Srivastava and Russ R Salakhutdinov. Multimodal Learning with Deep Boltzmann Machines. In *NeurIPS*, 2012.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. VL-BERT: Pre-training of Generic Visual-Linguistic Representations. In *ICLR*, 2020.

Chen Sun, Fabien Baradel, Kevin Murphy, and Cordelia Schmid. Learning Video Representations using Contrastive Bidirectional Transformer. *arXiv preprint arXiv:1906.05743*, 2019a.

Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. VideoBERT: A Joint Model for Video and Language Representation Learning. In *ICCV*, 2019b.

Hao Tan and Mohit Bansal. LXMERT: Learning Cross-Modality Encoder Representations from Transformers. In *EMNLP-IJCNLP*, 2019.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Deep Image Prior. In *CVPR*, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *NeurIPS*, 2017.

Dejing Xu, Jun Xiao, Zhou Zhao, Jian Shao, Di Xie, and Yueting Zhuang. Self-supervised Spatiotemporal Learning via Video Clip Order Prediction. In *CVPR*, 2019.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In *NeurIPS*, 2019.

Xiangyu Zhang, Xinyu Zhou, Mengxiao Lin, and Jian Sun. ShuffleNet: An Extremely Efficient Convolutional Neural Network for Mobile Devices. In *CVPR*, 2018.

Hang Zhao, Chuang Gan, Andrew Rouditchenko, Carl Vondrick, Josh McDermott, and Antonio Torralba. The Sound of Pixels. In *ECCV*, 2018.

Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason J Corso, and Jianfeng Gao. Unified Vision-Language Pre-Training for Image Captioning and VQA. In *AAAI*, 2020.

Linchao Zhu and Yi Yang. ActBERT: Learning Global-Local Video-Text Representations. In *CVPR*, 2020.

A Implementation Details

A.1 Architectures of Visual/Audio CNNs

Table 3 shows the architectures of visual and audio CNNs we use for our model. For the visual CNN, we use the SlowFast network (Feichtenhofer et al., 2019) with a ResNet-50 backbone (He et al., 2016). We use the speed ratio $\alpha = 8$ and the channel ratio $\beta = 1/8$ for the SlowFast architecture, so $T_f = 8 \times T_s$. We use different values of T_s and T_f for different tasks. During pretraining, we set $T_s = 4$ and $T_f = 32$. During finetuning, we use $T_s = 8$ and $T_f = 64$ for short-video action

classification on UCF101 (Soomro et al., 2012) while we use $T_s = 4$ and $T_f = 32$ for long-video action classification on Charades (Sigurdsson et al., 2016) and Kinetics-Sounds (Arandjelovic & Zisserman, 2017). For the audio CNN, we use a ResNet-50 without the downsampling layer pool_1 to preserve information along both frequency and time axis in early stages. We use different values of T_a for different training phases. We set $T_a = 220$ for one-second clip during pretraining while we use $T_a = 440$ for two-second clip during finetuning.

Stage	Visual CNN		Audio CNN
	Slow pathway	*Fast* pathway	
raw clip	$3 \times T_s \times 112^2$	$3 \times T_f \times 112^2$	$128 \times T_a$
conv_1	$1 \times 7^2, 64$ stride $1, 2^2$	$5 \times 7^2, 8$ stride $1, 2^2$	$9 \times 9, 32$ stride $1, 1$
pool_1	1×3^2, max stride $1, 2^2$	1×3^2, max stride $1, 2^2$	–
res_2	$\begin{bmatrix} 1 \times 1^2, 64 \\ 1 \times 3^2, 64 \\ 1 \times 1^2, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 3 \times 1^2, 8 \\ 1 \times 3^2, 8 \\ 1 \times 1^2, 32 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 32 \\ 3 \times 3, 32 \\ 1 \times 1, 128 \end{bmatrix} \times 3$
res_3	$\begin{bmatrix} 1 \times 1^2, 128 \\ 1 \times 3^2, 128 \\ 1 \times 1^2, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 3 \times 1^2, 16 \\ 1 \times 3^2, 16 \\ 1 \times 1^2, 64 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 4$
res_4	$\begin{bmatrix} 3 \times 1^2, 256 \\ 1 \times 3^2, 256 \\ 1 \times 1^2, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 3 \times 1^2, 32 \\ 1 \times 3^2, 32 \\ 1 \times 1^2, 128 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 6$
res_5	$\begin{bmatrix} 3 \times 1^2, 512 \\ 1 \times 3^2, 512 \\ 1 \times 1^2, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 3 \times 1^2, 64 \\ 1 \times 3^2, 64 \\ 1 \times 1^2, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 3$

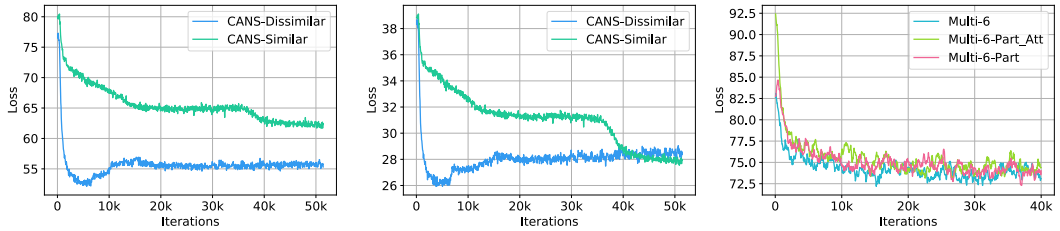
Table 3: The architectures of visual and audio CNNs. For the visual CNN, the input dimensions are denoted by *{channel size, temporal size, spatial size^2}*, kernels are denoted by *{temporal size, spatial size^2, channel size}* and strides are denoted by *{temporal stride, spatial stride^2}*. For the audio CNN, the input dimensions are denoted by *{frequency size, temporal size}*, kernels are denoted by *{frequency size, time size, channel size}* and strides are denoted by *{frequency stride, temporal stride}*.

A.2 Data Preprocessing

We preprocess the data by dividing T-second clips into T non-overlapping parts ($T = 30$ for Kinetics-700 (Carreira et al., 2019) and $T = 10$ for AudioSet (Gemmeke et al., 2017)) and sampling 16 frames from each. For audio stream, we take waveform sampled at 44.1 kHz and convert it to log-mel-scaled spectrogram. We augment audio data with random frequency/time masking using SpecAugment (Park et al., 2019), and visual data with color normalization, random resizing, random horizontal flip, and random cropping to obtain 112×112 pixel frames; for test data, we resize videos to 128 pixels on the shorter side and take three equidistant crops of 128×128 pixels to cover the entire region. We also apply audio-visual synchronized temporal jittering (Patrick et al., 2020).

A.3 Downstream Evaluation

For evaluation on UCF101, we follow the test protocol of (Feichtenhofer et al., 2019): We sample 10 clips from each test video at a uniform time interval, and for each sampled clip, we take three equidistant spatial crops, resulting in a total of 30 views. We use each of the 30 views as input to our visual CNN and average the prediction scores from all 30 views to obtain the final prediction result. For evaluation on ESC-50 (Piczak, 2015a), we extract 10 equally spaced 2-second clips from each test audio sample. We use each of 10 clips as input to our audio CNN and average the prediction scores to obtain the final prediction result. For evaluation on Charades and Kinetics-Sounds, we use three audio-visual sequences with different spatial crops from a test video and max-pool/average the prediction scores from each sequence, respectively.



(a-1) MEP + CPP (a-2) Visual MEP MEP + CPP

(a) Ablation: negative sampling (b) Ablation: parameter sharing

Figure 3: Loss curves during pretraining under different ablative settings. **(a)** compares Content-Aware Negative Sampling (CANS) that favors negatives that are dissimilar vs. similar to the positive instance. **(b)** compares different cross-Transformer weight sharing schemes; see the text for details.

A.4 Optimization

In all experiments, we use the AMSGrad (Reddi et al., 2018) variant of AdamW (Loshchilov & Hutter, 2019) optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.98$, L2 weight decay of 1e-4. We use a learning rate warm-up for the first 6% of iterations followed by a linear decay of learning rate. We pretrain our model on Kinetics-700 (Carreira et al., 2019) with a batch size 256 for 220K iterations and AudioSet (Gemmeke et al., 2017) with a batch size 300 for 220K iterations in the main experiments; for the ablation study, we use a much smaller batch size of 4 and pretrain our model on Kinetics-700 for 80K iterations. From the observations of Lezcano-Casado and Martínez-Rubio (Lezcano-Casado & Martínez-Rubio, 2019), we have 10 times less learning rate for the orthogonal parameters than that for the non-orthogonal parameters: we use 1e-5 for the former and 1e-4 for the latter. For finetuning on UCF101, we train our model for 40K iterations with a batch size of 64 and learning rate of 0.02. For finetuning on ESC-50, we train our model for 10K iterations with a batch size of 128 and learning rate of 0.02. For finetuning on Charades, we train for 40K iterations with a batch size of 8, with learning rate of 0.001 for the classifier and CNN parameters, 1e-5 for the orthogonal parameters and 1e-4 for the rest parameters. For finetuning on Kinetics-Sounds, we train for 24K iterations with a batch size of 32, with learning rate of 0.005 for the classifier and CNN parameters, 1e-4 for the orthogonal parameters and 1e-3 for the rest parameters.

B Extra Results from the Ablation Study

B.1 Negative Sampling Strategies

We proposed the content-aware negative sampling strategy (CANS) using pairwise l_2 distances between CNN embeddings. We introduced two variants of CANS: `CANS-Dissimilar` that favors negatives that are dissimilar to the positive instance and `CANS-Similar` that favors negatives that are similar to the positive instance. We chose to use `CANS-Similar` based on the results from our ablation study presented in the main paper, Table 1 (b).

Figure 3 (a) in this appendix provides additional evidence that supports our decision. We see that the loss of `CANS-Disimilar` initially drops rapidly but starts increasing around iteration 7K and continues to increase until around 15K; this is mainly caused by the visual MEP loss shown in Figure 3 (a-2). One explanation for this might that `CANS-Disimilar` is easier to solve than `CANS-Similar`, which causes the loss landscape of `CANS-Disimilar` to contain too many shallow local minima compared to that of `CANS-Similar`. Recall that we use a learning rate warm-up for the first 6% of iterations during pretraining; this roughly equals to the first 13K (out of 220K) iterations. Given this, we speculate that the model got out of a local minima around iteration 7K (most likely due to the increasing learning rate), and then eventually settled in another (bad) local minima after the warm-up period ended. Compared to this, we observe much milder learning dynamics with `CANS-Similar`: the loss decreases relatively slowly but steadily, and eventually leaps around 35K to go below the loss of `CANS-Disimilar`. We, again, believe that this is because `CANS-Similar` is more difficult to solve than `CANS-Disimilar` (as shown by the slower decrease in loss values), which caused the resulting loss landscape to contain steeper local

minima. Our model eventually found one of those after round 40K of iterations, resulting in a better performing model in the downstream tasks shown in Table 1 (b) of the main paper (the loss kept slowly decreasing after iteration 50K).

B.2 Parameter Sharing Schemes

We proposed a cross-Transformer weight sharing technique, which decomposes weight matrices with low-rank approximation. Recall that each layer of a Transformer contains the multi-head attention layer weights $\{W^q, W^k, W^v, W^b\}$ and the feed-forward layer weights $\{W^c, W^d\}$. We chose to share all six weight matrices across Transformers, though we could have shared any combination of them. To justify this design choice, we empirically compared three variants: (i) `Multi-6` that do not share parameters across Transformers, (ii) `Multi-6-Part_Att` that shares only $\{W^q, W^k, W^v, W^b\}$ (but not $\{W^c, W^d\}$) and (iii) `Multi-6-Part` that shares all six weight matrices. Figure 3 (b) shows that there is not much difference between all the variants in terms of the loss curves; we chose to use `Multi-6-Part` that requires the least number of parameters. We showed that our approach outperforms `Multi-6` in the ablation study (Table 1 (c-left) in the main paper).

B.3 Justification for the MEP Loss Formulation

Since the multimodal Transformer h_{AV} has access to both visual and audio inputs, one might think that the model could "leak" information about visual input into $\mathbf{z}^a$ and information about audio input into $\mathbf{z}^v$, which could make MEP trivial to solve. Here we show that this is not the case. By construction, we mask the same positions in audio and visual streams when designing the MEP task, so the model has no access to the masked input even in a cross-modal manner. Empirically, removing the third term in Eq. 6 ($\mathcal{L}_{\text{NCE}}([\mathbf{x}^a; \mathbf{x}^v], \tilde{\mathbf{z}})$) leads to performance degradation in Kinetics-Sounds, i.e., top-1 accuracy 66.7% vs. ours 67.5% (see Table 1), which suggests that solving the MEP task in the multimodal Transformer is beneficial to our model.

B.4 Justification for the CPP Loss Formulation

Recall that our CPP loss has two terms; the first term uses the summary embeddings $\mathbf{s}_g$ and the second term uses output embeddings $\mathbf{s}_h$ sampled at random positions; see Eq. 7. One could argue that the two terms are redundant as bidirectional Transformers have "one-step" access to all the input embeddings, and thus solving CPP only with the summary embeddings (the first term) would be enough. This is not the case. We encode $\mathbf{s}_h$ with position-specific information through the time embeddings $\mathbf{p}_t$, which makes every $\mathbf{s}_h$ different compared to $\mathbf{s}_g$. Empirically, we find that removing the second term of Eqn. 7 ($\mathbf{s}_h$) in our CPP loss leads to an inferior accuracy 66.9% vs. ours 67.5% on Kinetics-Sounds, suggesting its importance in learning.

B.5 Use of Modality Embeddings in the Multimodal Transformer

We use modality embeddings $\mathbf{m}^v$ and $\mathbf{m}^a$ as part of input to the multimodal Transformer in order to distinguish embeddings coming from visual and audio Transformers. They are learnable weights trained end-to-end with other parameters. Conceptually, incorporating modality-discriminative embeddings is crucial because of our aggressive weight sharing scheme. Without them, the multimodal Transformer will see the output from audio/visual Transformers (y^a and y^v) as if they are coming from the same distribution because the two Transformers share a large part of weights. Using modality embeddings encourages our model to preserve modality-specific information in the final output, and this empirically leads to performance improvements: ours 67.5% vs. without modality embeddings 67.1% on Kinetics-Sounds.