Information Fusion 137 (2027) 104613

Contents lists available at ScienceDirect

Information Fusion

journal homepage: www.elsevier.com/locate/inffus

Full Length Article

# Audio-guided visual selection for efficient multimodal fusion via a discrete variational autoencoder

Hung Le [a,*], Hung-Hsuan Huang [b], Candy Olivia Mawalim [a], Chee Wee Leong [c], Shogo Okada [a,*]

[a] Japan Advanced Institute of Science and Technology, Nomi, Ishikawa, Japan
[b] The University of Fukuchiyama, Fukuchiyama, Kyoto, Japan
[c] Educational Testing Service, Princeton, New Jersey, USA

ARTICLE INFO

*Keywords:*
Deep multimodal learning
Discrete variational autoencoder
Inference efficiency
Salient segment selection
Interview video processing

ABSTRACT

In multimodal learning, not all parts of all modalities are of equal importance. An additional modality can introduce overlapping signals while requiring a significant increase in computation. As a result, there are many cases where computational requirements for a multimodal model are increasing while its performance is degrading. Early detecting and subsequently fusing only the complementary segments that exist in the high-resource modality by previewing the low-resource modality is how one can overcome this problem. This work proposes a framework to dynamically learn the selection of such salient segments in the vision feature space by previewing the acoustic features of the audio. The framework contains a self-supervised discrete variational autoencoder (dVAE) component and a traditional supervised multimodal fusion component. The former is trained to produce a discrete masked vector in the latent space, as opposed to the traditional continuous latent VAE, by using Binary Concrete distributions as the encoder and the prior distributions. This masked vector removes the non-salient segments in the vision feature space before passing them to the latter component. At inference time, only salient vision features need to be extracted, making the framework very efficient. The empirical experiment on a large-scale video interview dataset reveals that this approach outperforms SOTA in terms of both efficiency and performance, requiring only 15.42 percent of the original vision features (a 23% relative improvement) while improving correlation from 0.6792 to 0.7196 and F1-score from 74.73% to 76.64%.

## 1. Introduction

We digitalize our complex world via various input devices such as microphones, cameras, sensors, etc. Multiple modalities (i.e., views) of an event can be captured simultaneously by these devices. The field of multimodal learning aims to improve the robustness and generalizability of artificial intelligence by fusing complementary information from these modalities [1]. In some ideal scenarios and real-world applications, the additional supply of abundant information will improve the performance of multimodal models compared to their unimodal model counterparts [2,3]. In practice, efficiently learning from multimodal data in the era of large-scale deep learning remains a major challenge. It is neither efficient nor feasible for a model to learn from all parts of all modalities at scale due to computational constraints [4]. This problem cannot be easily resolved by adding computational resources as some of the modalities are dense and high-dimensional, leading to a low signal-to-noise ratio for a particular task, making the whole multimodal learning approach counterproductive. Additionally, a less mentioned but increasingly important concern is the cost of inference [5,6]. The deployment cost of multimodal models is high due to the additional resource requirements (e.g., GPUs, electricity) and the opportunity cost as the time from the user's first request to the system's first response increases. In this paper, we formalize these problems and propose our solution to the question: *Given a set of input modalities, how to select the most salient complementary segments and improve the inference efficiency of multimodal models?*

In search of the answer, we first must ask ourselves, what is an *efficient model*? In the context of this paper, an efficient model is a model that uses the *right* amount of computation for an input sample while maintaining the performance requirements. The human brain, conceptually, is a very efficient model as most of the input from the five senses is discarded and only informative segments are processed. Furthermore, the brain allocates a different amount of cognitive power for processing the task at hand. Compact models, on the other hand, aim to use the *least*

* Corresponding authors.
*E-mail addresses:* hungle@jaist.ac.jp (H. Le), okada-s@jaist.ac.jp (S. Okada).

https://doi.org/10.1016/j.inffus.2026.104613
Received 3 March 2026; Received in revised form 10 June 2026; Accepted 8 July 2026
Available online 11 July 2026

amount of computation averaged from the sample distribution. They are generally trained using techniques such as knowledge distillation, quantization, pruning, or low-rank adaptation [7]. While the developed techniques have achieved tremendous success, the one-size-fits-all compact models still use the same amount of computation for every input they receive. Part of the reason is that deep learning hardware and frameworks are optimized to process batches of samples in parallel by requiring them to have equal length. At training time, this pipeline is very effective as all samples are available to be divided into batches. At inference time, however, the samples do not come in batches but rather one by one. If a model has the ability to dynamically reduce or increase the computation needed per individual sample during inference, it would then be very efficient, especially when multimodalities are involved. In this spirit, building on previous work [8,9], we further explore the problem of constructing efficient multimodal models by dynamically allocating computational resources for multimodal processing.

This problem is an important one due to the implications of its solutions. As machine learning systems are deployed at a massive scale, efficient models will reduce energy consumption, thereby saving both operational costs and the environment. In the case of human-centric video understanding, resource requirements scale exponentially with the introduction of each additional modality [10]. The reason lies in how different modalities are represented (e.g., language as 1D vectors, audio as 2D matrices, images as 3D tensors, and video as 4D tensors) and their combination. An inefficient multimodal model not only requires significantly more computation but it is also ineffective, as useful signals in one modality can be diluted by the noise introduced by another [11]. Following this reasoning, efficient models have the potential to achieve performance improvements while avoiding the degradation commonly observed in compact architectures.

Several previous works have attempted this problem. One way to design efficient models is to follow the coarse-to-fine framework [12]. Under this framework, a lightweight model is trained with coarse features to decide whether the current frame is worthy of fine feature inspection or should be skipped. We recognize two shortcomings of this approach. First, the coarse features of all frames still need to be extracted. Second, we know that the modalities capture the same event and that different modalities require different amounts of computation, but these prior knowledge is not incorporated. Another way to design efficient models was proposed by [13], where a video is trimmed into several clips (i.e., segments) and only the first image and the audio of the clips are used to train the model. A skimming module then decides which of the key moments (a subset of image-audio pairs) to be used for video-level action recognition. As images and audio are much less compute-intensive than the whole video itself, the resulting model is more efficient. While this method works well for action recognition tasks, a problem arises when we consider the situation where a short sequence of frames contains important semantic information. This situation frequently occurs in affective computing: changes in facial expressions are captured in a short clip, and a single frame will fail to provide the necessary information to correctly understand the expression. Using only one frame to represent a segment during the training phase forces the model to learn with missing data, which is suboptimal. The ideas from these previous works and their remaining limitations motivate us to propose EMF-dVAE (Efficient Multimodal Fusion with discrete Variational Autoencoder), a framework that addresses these challenges by processing all vision features during training while using only salient vision features during inference.

EMF-dVAE was designed for human-centric video understanding tasks, where typical modalities are language, audio, and vision. The two main components of EMF-dVAE are a discrete variational autoencoder (dVAE) and a multimodal fusion (MF) network. The discrete variational autoencoder can be trained in a self-supervised manner. This component aims to reconstruct the vision feature space from the corrupted vision features, where the corrupted segments are determined by the audio features. The resulting encoder of the dVAE can output a binary mask vector in the latent space that marks the salient segments. Since the sampling process of a binary vector from a discrete distribution breaks the differentiability requirements of backpropagation, we replace discrete distributions with Binary Concrete distributions, which enable differentiable sampling. During training, dVAE has access to all available vision features to learn what is important and what is not. At inference time, only the encoder of the dVAE is activated and only features of salient vision segments are extracted and passed to the MF component. This design choice reflects different cost considerations: training occurs once and should leverage all available data (vision features can be pre-extracted and reused across epochs), while inference happens repeatedly in deployment and must prioritize efficiency (extracted vision features are used only once). The second component of EMF-dVAE, the MF network, simply contains a fusion operator and a deep multimodal network. While we maintain a simple design for this component to aid the evaluation of mask vectors produced by dVAE, future work can explore more sophisticated architectures.

For evaluation, we conducted experiments on a large-scale video interview dataset. The empirical results suggest that EMF-dVAE is not only more efficient than traditional approaches but also more effective, establishing new state-of-the-art (SOTA) results. We conducted analysis and ablation studies to further validate this finding. In summary, the main contributions of this paper are:

- We formalize the problem of efficiency in multimodal models during the inference phase.
- We propose the EMF-dVAE framework to tackle this problem. The core contribution is a novel audio-guided selective fusion mechanism, in which a VAE with Binary Concrete distribution learns to generate a binary mask vector that identifies and retains only the most salient visual segments for the downstream prediction task.
- We conduct experiments on a large-scale video interview dataset. The experimental results suggest that the framework is robust and efficient.

## 2. Preliminaries

The idea of previewing the audio modality and deciding which vision segments to invest resources in requires discrete decisions. The backpropagation algorithm, which powered the success of neural networks, on the other hand, is only sufficient when the loss function is a deterministic and differentiable function of the parameter vector. In this section, we present the mathematical background previously developed for training neural networks with discrete and non-differentiable layers, for their importance in a rich class of problems, including ours. Additionally, we review the original variational autoencoders framework and their training objective as they served as the backbone of the dVAE component. The last part of the section discusses a specific challenge in multimodal learning when only one label is available for long-form video or time-series data.

### 2.1. Backpropagation through discrete stochastic nodes

A stochastic computation graph is a directed acyclic graph that contains stochastic nodes in addition to deterministic nodes (for a rigorous definition, see [14]). A neural network is non-differentiable due to these stochastic nodes. The stochastic nodes are distributed conditionally on their parents; in other words, these non-deterministic nodes are sampled from a conditional distribution. Consider an input $\mathbf{x}$ and a stochastic node $S$ in a stochastic computation graph. Denote $Pa(S)$ as the parent of the stochastic node $S$. In the forward pass, we first sample $S$ from the conditional distribution $p_\phi(S|\mathbf{x}, Pa(S))$ and evaluate a deterministic loss function $f_\theta(x)$ at $S$. To avoid clutters in notation, we denote $p_\phi(S|\mathbf{x}, Pa(S))$ as $p_\phi(\mathbf{x})$. Treating $f_\theta(S)$ as a noisy objective, we would like to optimize its expected value $\mathcal{L}(\theta, \phi) = \mathbb{E}_{S\sim p_\phi(x)}\left[f_\theta(S)\right]$ w.r.t. parameters $\theta, \phi$. The objective and its gradients are intractable; however, we can estimate them with samples from $p_\phi(x)$.

The gradient of the expected value w.r.t. $\theta$ is:

$$\begin{aligned}\nabla_\theta \mathcal{L}(\theta,\phi) &= \nabla_\theta \mathbb{E}_{S\sim p_\phi(x)}\left[f_\theta(S)\right] \\ &= \mathbb{E}_{S\sim p_\phi(x)}\left[\nabla_\theta f_\theta(S)\right]\end{aligned} \tag{1}$$

and can be estimated with Monte Carlo sampling:

$$\nabla_\theta \mathcal{L}(\theta,\phi) \simeq \frac{1}{K}\sum_{k=1}^{K}\nabla_\theta f_\theta(S_k) \tag{2}$$

where $S_k \sim p_\phi(x)$ *independently and identically distributed* (i.i.d).

The gradient of the expected value w.r.t. the distribution parameters $\phi$ is more challenging to compute:

$$\begin{aligned}\nabla_\phi \mathcal{L}(\theta,\phi) &= \nabla_\phi \mathbb{E}_{S\sim p_\phi(x)}\left[f_\theta(S)\right] \\ &= \nabla_\phi \int p_\phi(x) f_\theta(x)dx \quad \text{(LOTUS rule)} \\ &= \int f_\theta(x)\nabla_\phi p_\phi(x)dx \quad \text{(Leibniz's rule)}\end{aligned} \tag{3}$$

This gradient does not lead to a Monte Carlo gradient estimator because it does not have the form of an expectation w.r.t. $x$.

### 2.1.1. The score function estimator

The score function estimator (also known as REINFORCE or likelihood ratio estimator) uses the identity $\nabla_\phi p_\phi = p_\phi(x)\nabla_\phi \log p_\phi(x)$ to transform the gradient in Eq. 3 to:

$$\begin{aligned}\nabla_\phi \mathcal{L}(\theta,\phi) &= \int f_\theta(x) p_\phi(x)\nabla_\phi \log p_\phi(x)dx \\ &= \mathbb{E}_{S\sim p_\phi(x)}\left[f_\theta(S)\nabla_\phi \log p_\phi(S)\right]\end{aligned} \tag{4}$$

The identity serves two purposes: (1) to transform Eq. 3 into an expectation involving the derivative of the probability and (2) to replace the probability with the log probability. The log probability formulation (2) tends to be more numerically stable for floating-point calculation on computers, and the expectation w.r.t. $x$ formulation (1) allows us to use the Monte Carlo gradient estimator:

$$\nabla_\phi \mathcal{L}(\theta,\phi) = \frac{1}{K}\sum_{k=1}^{K} f_\theta(S_k)\nabla_\phi \log p_\phi(S_k) \tag{5}$$

where $S_k \sim p_\phi(x)$ (i.i.d.). This estimator requires that it is possible to sample from $\log p_\phi(x)$ and $\log p_\phi(x)$ is differentiable w.r.t. $\phi$. In practice, we can choose a distribution $p_\phi(x)$ that satisfies both requirements. While this estimator is very general since $f_\phi(x)$ does not need to be differentiable or continuous as a function of $x$, its high variance disallows efficient and stable training [15,16].

### 2.1.2. The reparameterization trick

The high variance of the score function estimator motivates the development of the *reparameterization trick* as an alternative path to overcome the sampling problem in Eq. 3. This trick was first introduced in the context of variational inference [17]. In Eq. 3, instead of sampling $S$ from $p_\phi(\mathbf{x})$, we calculate $S$ using a deterministic function $g_\phi(z)$: $S = g_\phi(Z)$ where $Z \sim q(z)$. This step has moved the stochastic process from $S$ to $Z$. For example, in many cases, we want $S$ to follow a Gaussian distribution. We can define the function $g_\phi$ as $g_{\mu,\sigma}(\epsilon) = \mu + \sigma\epsilon$ where $\epsilon \sim N(0,1)$, then sampling $S$ from a Gaussian distribution (with mean $\mu$ and standard deviation $\sigma$) is equivalent to sampling $\epsilon$ from a standard Gaussian distribution and determine $S = g_{\mu,\sigma}(\epsilon)$ (due to the fact that all Gaussians are affine transformations of the standard Gaussian distribution).

Assume that $f_\theta(x)$ is differentiable w.r.t. $x$ and $g_\phi(z)$ is differentiable w.r.t. $\phi$, substitute $X = g_\phi(Z)$ to Eq. 3, we have:

$$\begin{aligned}\nabla_\phi \mathcal{L}(\theta,\phi) &= \nabla_\phi \mathbb{E}_{X\sim p_\phi(x)}\left[f_\theta(X)\right] \\ &= \nabla_\phi \mathbb{E}_{Z\sim q(x)}\left[f_\theta(g_\phi(Z))\right] \\ &= \mathbb{E}_{Z\sim q(x)}\left[\nabla_\phi f_\theta(g_\phi(Z))\right] \\ &= \mathbb{E}_{Z\sim q(x)}\left[f'_\theta(g_\phi(Z))\nabla_\phi g_\phi(Z))\right] \quad \text{(Chain rule)}\end{aligned} \tag{6}$$

As shown in Eq. 6, the reparameterization trick reduces the problem of estimating the gradient w.r.t. parameters of a distribution to the simpler problem of estimating the gradient w.r.t. parameters of a deterministic function.

### 2.1.3. The Gumbel-Softmax or concrete estimator

In some situation, we would like $S$ to follow a discrete (categorical) distribution $p_\pi(x)$, where $\pi = (\pi_1, \ldots, \pi_k)$ and $\pi_i \in (0,1)$ are the class probabilities. The Gumbel-Max trick [18] takes advantage of the reparameterization trick by constructing the deterministic function $g_\pi(z)$ in the following manner:

$$\begin{aligned}g_\pi(z_i) &= \arg\max_i \left[z_i + \log(\pi_i)\right] \\ &= \arg\max_i \left[-\log\left(-\log(U_i)\right) + \log(\pi_i)\right]\end{aligned} \tag{7}$$

where $U_i \sim$ Uniform(0, 1) i.i.d. for each $i$ and $z$ is substitute by $-\log\left(-\log U_i\right)$ which is a sample from the Gumbel distribution. The Gumbel distribution is chosen for the added noise since it is stable under the max operations. Since arg max is a discrete and non-differentiable function, the softmax function is used as its continuous and differentiable approximation in the Gumbel-Softmax estimator:

$$y_i = \frac{\exp\left(\left(\log\left(\pi_i\right) + z_i\right)/\tau\right)}{\sum_{j=1}^{k}\exp\left(\left(\log\left(\pi_j\right) + z_j\right)/\tau\right)} \tag{8}$$

where $y_i$ is the value at index $i$ in the one-hot vector of size $\mathbb{R}^k$. In Eq. 8, $\tau$ is a temperature parameter to control the "softness" of the distribution. When $\tau \to 0$, samples from the Gumbel-Softmax distribution are identical to samples from a categorical distribution. In the literature, the Gumbel-Softmax distribution is also often referred to as the Concrete distribution [19,20].

### 2.1.4. The relaxed Bernoulli or binary concrete distribution

The relaxed Bernoulli or Binary Concrete distribution is a special case of the Gumbel-Softmax distribution. To sample $S$ from BinConcrete($\pi$), where $\pi$ is the probability of success, we can sample $L \sim$ Logistic($\mu = 0, s = 1$) and set

$$S = \frac{1}{1+\exp(-(\log\alpha + L)/\tau)} = \text{Sigmoid}\left(\frac{\log\alpha + L}{\tau}\right) \tag{9}$$

where $\log\alpha = \log(\pi/(1-\pi))$. $\pi/(1-\pi)$ is the ratio of success probability to failure probability for a Bernoulli distribution, and by varying the value of $\pi$, we can impose our prior bias. In Eq. 9, $\log\alpha$ is the only variable that carries information about which class (0 or 1) is more likely. As the range of $\log\alpha$ is all real numbers, $\log\alpha$ is implemented as the pre-activation output of a neural network. The reparameterization trick allows the gradients to flow through the BinConcrete sampling process as the source of randomness is delegated to the Logistic sampling. For the definition and properties of the Binary Concrete random variable, we refer readers to Appendix B in [20]. For the rest of this paper, we will refer to this distribution as the Binary Concrete distribution.

### 2.1.5. The straight-through trick

A useful trick when training end-to-end models with Concrete distribution is the straight-through trick. To use this trick, we simply discretize the samples from the Concrete distribution during the forward pass of the network using a predefined threshold, while using the "soft" values in the backward pass for backpropagation.

## 2.2. Variational autoencoders

Variational autoencoder (VAE, [17]) is a powerful framework for unsupervised learning of latent variable representations. Consider a dataset $\mathcal{D}$ of $N$ i.i.d. observations:

$$\mathcal{D} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(N)}\} \tag{10}$$

where $\mathbf{x}$ are observed variables. We assume that there is a probability distribution from which the data were sampled. Under the i.i.d. assumption, the log-probability assigned to the data by the model is:

$$\log p_\theta(D) = \log\left(\prod_{\mathbf{x}\in D} p_\theta(\mathbf{x})\right) = \sum_{\mathbf{x}\in D} \log p_\theta(\mathbf{x}) \tag{11}$$

Probabilistic models aim to learn the probability distribution $p_\theta$ usually via *maximum likelihood estimation* (MLE). In other words, we try to find the set of parameters $\theta$ of a distribution $p$ that will maximize the probability $p_\theta(D)$. The log-probability is used instead of probability to handle extreme values and simplified calculations while does not affect the behavior of the probability function. In addition to observed variables $\mathbf{x}$ from the dataset, it is often the case that there are unobserved variables $\mathbf{z}$ which are a part of the model. As $\mathbf{z}$ are unobservable (or even definable), they are called *latent variables*. VAE can be viewed as the process of finding the distribution that maps the observable variables $\mathbf{x}$ to the latent variables $\mathbf{z}$ such that the latent representations can capture the essence of the original data ($\mathbf{z}$ is meaningful for data generation). To achieve this goal, VAE was designed with two coupled but independently parameterized models: the encoder and the decoder.

#### 2.2.1. The encoder and decoder

To properly understand the encoder of VAE, we first discuss *deep latent variable models* (DLVM). A DLVM $p_\theta(\mathbf{x},\mathbf{z})$ is a latent variable model whose distributions are parameterized by neural networks. The marginal distribution over the observed variables $p_\theta(\mathbf{z})$ is then given by:

$$p_\theta(\mathbf{x}) = \int p_\theta(\mathbf{x},\mathbf{z}) d\mathbf{z} \tag{12}$$

This marginal probability is typically intractable, therefore the conditional probability:

$$p_\theta(\mathbf{z}|\mathbf{x}) = \frac{p_\theta(\mathbf{x},\mathbf{z})}{p_\theta(\mathbf{x})} \tag{13}$$

is also intractable. To overcome this problem, VAE introduces an inference model (the encoder) $q_\phi(\mathbf{z}|\mathbf{x})$ and optimizes the variational parameters $\phi$ such that $q_\phi(\mathbf{z}|\mathbf{x}) \approx p_\theta(\mathbf{z}|\mathbf{x})$. In practice, the encoder is parameterized by a neural network ($\phi$ include the weights and biases of the network), for example:

$$\begin{aligned}(\mu, \log(\sigma)) &= \text{EncoderNeuralNet}_\phi(\mathbf{x})\\ q_\phi(\mathbf{z}|\mathbf{x}) &= \mathcal{N}(\mathbf{z}; \mu, \text{diag}(\sigma))\end{aligned} \tag{14}$$

The generative model (the decoder) $p_\theta(\mathbf{x}|\mathbf{z})$ is another model that learn the mapping from $\mathbf{z}$ back to $\mathbf{x}$. Theoretically, the decoder also outputs the parameters of $\theta$, but typical implementation of the decoder outputs the sample directly, which is interpreted as the mean sample of the Gaussian distributions.

#### 2.2.2. Evidence lower bound

VAE are trained based on the principle of MLE [21]. We have:

$$\begin{aligned}\log p_\theta(\mathbf{x}) &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x})\right]\\ &= \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log\left[\frac{p_\theta(\mathbf{x},\mathbf{z})}{q_\phi(\mathbf{z}|\mathbf{x})}\right]\right] + \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log\left[\frac{p_\theta(\mathbf{z}|\mathbf{x})}{q_\phi(\mathbf{z}|\mathbf{x})}\right]\right]\end{aligned} \tag{15}$$

The second term in Eq. 15 is the KL-divergence of $p_\theta(\mathbf{z}|\mathbf{x})$ from $q_\phi(\mathbf{z}|\mathbf{x})$. Since the KL-divergence is non-negative, maximizing only the first term will maximize $\log p_\theta(\mathbf{x})$. The first term is therefore known as the *evidence lower bound* (ELBO):

$$\mathcal{L}_{\theta,\phi}(\mathbf{x}) = \mathbb{E}_{q_\theta(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x},\mathbf{z}) - \log q_\phi(\mathbf{z}|\mathbf{x})\right] \tag{16}$$

which can be regrouped as:

$$\mathcal{L}_{\theta,\phi}(\mathbf{x}) = \underbrace{-\text{KL}\left[q_\phi(\mathbf{z}|\mathbf{x}) \,||\, p_\phi(\mathbf{z})\right]}_{\text{KL-divergence}} + \underbrace{\mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p_\theta(\mathbf{x}|\mathbf{z})\right]}_{\text{reconstruction loss}} \tag{17}$$

In practice, the reconstruction loss, $\mathcal{L}_{\text{reconstruction}}$, could be (but not limited to) the mean square error loss or the cross entropy loss. Additionally, since neural networks are optimized under the principle of gradient descent (i.e, minimizing the loss function), the loss function of VAE becomes:

$$\mathcal{L}_{VAE} = \text{KL-divergence} + \mathcal{L}_{\text{reconstruction}} \tag{18}$$

### 2.3. Multimodal representation learning from video-level annotations

We are now discussing a more practical and specific challenge in multimodal representation learning. In human-centric video understanding, there are two main ways to collect the annotations. The first way is to collect the labels at the *frame-level*. In this case, annotators annotate each frame or short sequence of frames (up to a few seconds) with the corresponding label. This type of annotation is common for basic perception recognition tasks such as emotion recognition or engagement prediction. This process is expensive, time-consuming, and only feasible for small-scale datasets. The second way to annotate the labels is at the *video-level*. Each video (with a length of a few minutes) is watched uninterruptedly by the annotator, and a label is assigned to the whole video. This type of annotation is suitable for advanced cognitive reasoning tasks where the annotators are required to be domain experts. Some examples of *video-level* annotations are speech essay rating or hirability assessment. While the annotations are not as expensive to obtain, *video-level* annotated tasks pose their own challenges.

We argue that the main challenge is the discrepancy between the input multimodal data and the prediction task. It is often not possible even for the annotators to describe exactly why a decision for a score is made, as the multimodal signals are processed unconsciously in the brain. A single output label provides too little feedback for a model to learn the connection between the input videos and the final ratings. In unimodal learning, this discrepancy problem has been addressed by moving from supervised learning to self-supervised learning. A model is trained unsupervised on a large-scale dataset to learn general latent features, and these features are transferred to the supervised learning task. In multimodal learning, unsupervised learning of the fused modality representations remains a difficult problem. To combat this challenge, this work combines a self-supervised component to denoise unimodal representations with a supervised component for final prediction.

## 3. Method

We start this section by formulating the efficiency problem in multimodal processing during inference and then describe the proposed framework.

### 3.1. Problem statement

In this work, we focus on human-centric video understanding, which is one of the most common types of multimodal processing. Typically, a video can be decomposed into three separate modalities: vision (V), audio (A), and language (L). Vision refers to the frames (images) in the video, audio refers to the sound waves captured as the electric signal, and language refers to the spoken content.

The prediction task is simple to describe: given a video, our goal is to construct a model that can place a label on the video. The label is the answer to a particular question of interest, e.g., *Were the person in the video happy or sad? (emotion recognition task)* or *Should we consider hiring this person? (hirability assessment task)*. To train such models, the common approach is to follow a two-stage process: feature extraction and model training. In the feature extraction stage, feature vectors for each modality are extracted using a learnt encoder. These encoders are typically pre-trained on the modality-specific data to extract high-level representations from raw inputs. For example, the OpenFace toolkit [22], which contains several facial (CNN-based) encoders,

is commonly used for feature extraction. The extracted action units then serve as input vision features in the model training stage. In the context of this work, we denote the extracted vision, audio, and language features as $X_{\{v,a,l\}} \in \mathbb{R}^{T_{\{v,a,l\}} \times d_{\{v,a,l\}}}$, where $T$ is the sequence length for each modality and $d$ is the corresponding feature dimension. In the model training stage, these pre-extracted features are used as input to develop multimodal models. Algorithm S1 and Algorithm S2 (Supplementary Material (SM)) show the two-stage process in more detail.

There are several advantages to this decoupling procedure. First, since the encoders in stage one were tailored to a specific modality, they excelled at extracting unimodal features. The models developed in stage two could then focus on how to effectively fuse information among the modalities to make the final prediction without having to worry about the unimodal representation learning task. Additionally, as the unimodal features for all samples were already available, sampling a mini-batch can be done efficiently. Without the pre-extracted features, we would not be able to take advantage of the parallel processing power of modern GPUs. Lastly, in many cases, the collectors of the datasets are not allowed to share the original video files since they may contain personally identifiable information elements. By sharing only the extracted features, this issue could be somewhat mitigated, and other research groups can still develop models from these datasets. The shared extracted features also provide the same unimodal starting point, making it fairer to compare different proposed multimodal approaches.

One pitfall of this procedure arises when we consider real-world inference scenarios. In such cases, the system does not receive a batch of videos, but rather one video at a time. The system also needs to execute both stages consecutively for one video to get a prediction. Our aim is to minimize the inference time - the time it takes for the system to receive a video and return a prediction to the user. One approach is to make the model $f$ in stage two more compact, which is the focus of some previous work. While it is known in multimodal learning that the contributions of modalities are imbalanced [23], the unimodal feature extraction step has not made use of this fact. We viewed this as an opportunity to be more efficient and asked the question: Can we make the whole process more efficient during inference-time, when both stages need to be executed consecutively?

### 3.2. The proposed framework

#### 3.2.1. Overview

To answer the main research question, we propose using the audio modality to guide the selection of vision feature segments to be processed. As language and audio are considered "light" modalities, the complexity of their computation is within the acceptable realm. The proposal came from the observation that humans do not pay full attention to the visual cues at all times. Fig. 1 depicts the overview of the EMF-dVAE framework during training. The training starts with the extracted audio, vision, and language features. The key novelty of the EMF-dVAE is the introduction of the Masker model $g_\phi$ (i.e., the dVAE component). $g_\phi$ takes the audio features as its input and outputs a mask vector. The mask vector has the same size as the vision features, and we want it to be in the form of a binary vector. Assuming that the temporal dimension of the vision features is unchanged after the feature extraction process, we can mask out part of the vision modality with zeros by element-wise multiplying it with the mask vector. Conceptually, this procedure attempts to mimic the mechanism of human visual attention: we look when we hear something of interest. The second component, which we call the Predictor (i.e., the MF component), takes multimodal features as input, fuses and then passes them to a multimodal fusion network. This component is responsible for making the final predictions.

Note that once the Masker model is trained, there is no longer a need to extract all of the vision features from the whole video during inference. Instead, only features of non-masked segments are extracted, and the rest of the feature vector can be filled with zeros. Algorithm 1 formalizes this behavior of EMF-dVAE during inference.

**Algorithm 1** EMF-dVAE during inference.

**Require:** $N$ raw videos, each video can be decomposed into three raw data types: vision (V), audio (A), and spoken content (L).
**Require:** Pre-trained encoders or feature extractors $E_v$, $E_a$, $E_l$ for corresponding vision, audio, and language modalities.
Inference time, processing new video $k$:
Decompose the video $k$ into $V^k$, $A^k$, and $L^k$.
Extract audio features: $X_a^k = E_a\left(A^k\right)$
Extract language features: $X_l^k = E_l\left(L^k\right)$
Extract the mask vector from the masked model: $\mathsf{mask}^k = g_{\theta_1}\left(X_a^k\right)$
Mask the vision features: $\tilde{V}^k = \mathsf{mask}^k \odot V^k$
Extract masked vision features: $\tilde{X}_v^k = E_v\left(\tilde{V}^k\right)$
Combine into one video feature: $\tilde{X}_{v,a,l}^k \leftarrow \left(\tilde{X}_v^k, X_a^k, X_l^k\right)$
Make prediction: $\hat{y} = f_{\theta_2}\left(\tilde{X}_{v,a,l}^k\right)$

#### 3.2.2. The masker model

Our goal for the Masker model is to receive an audio segment as input and output a binary vector marking the most informative segments. The binary vector lies in $\mathbb{B}^{T_v}$, where $\mathbb{B}$ is the Boolean domain. This mask vector operates solely along the temporal dimension and not on the feature dimensions. The element-wise operation between $\mathsf{mask}^k \in \mathbb{B}^{T_v}$ and $V^k \in \mathbb{R}^{T_v \times d_v}$ is made possible by PyTorch's broadcasting operation. A function that outputs only binary vectors is not differentiable, hence cannot be a part of a neural network as is. One way to achieve this goal and overcome this obstacle is to train a neural network whose outputs represent parameters of Binary Concrete distributions ($\log \alpha$ or $\pi$ in Eq. (9)), then sample from these distributions independently to produce the mask vector. The properties of the Binary Concrete distribution allow each component of the mask vector to be close to 1 or 0, and the reparameterization trick (Eq. (6)) allows the gradient to flow through the stochastic sampling operation. To narrow the gap between training and inference, the straight-through trick is used to discretize 0s and 1s during the forward pass, while the approximations are used during the backward pass. With these modifications, the framework shown in Fig. 1 can be trained in an end-to-end manner, using only the target label. Unfortunately, there are two main pitfalls with this approach. First, the model has to use only one target label for the whole training sequence, so the objective function does not explicitly guide the model toward choosing the most representative segments. This is the challenge previously discussed in Section 2.3. Second, as the size of the binary vector increases linearly (in the case of longer videos or finer segmentations), the possible choices for the binary vector grow exponentially. As a result, it would take exponentially more runs for the model to try out different combinations for the mask vector. The infeasibility of this solution inspires us to resort to VAE.

Recall that one interpretation of VAE is to learn the deep latent variables $\mathbf{z}$ that compress the original data. The vector $\mathbf{z}$ is in the much simpler $\mathbf{z}$-space compared to the original $\mathbf{x}$-space. The success of VAE shows that it is possible to learn the distribution $q_\phi(\mathbf{z}|\mathbf{x})$ without having to try all possible combinations for $\mathbf{z}$. If we treat the mask vector as a sample drawn from $q_\phi(\mathbf{z}|\mathbf{x})$, then we can overcome the computational problem mentioned in the previous paragraph. Fig. 2 illustrates the modifications we made to the original VAE to learn the mask vector, highlighted in blue. At the framework level, the input to the encoder is one modality (audio) while the output of the decoder is another (vision). It is unreasonable to ask the model to reconstruct the vision features based solely on the audio features, so the input to the decoder is actually the masked version of the vision features (constructed using $\mathbf{z}$). In other words, the decoder's task is to reconstruct the original vision features given a subset of them, which is a reasonable task since many of the vision features are redundant. At the implementation level, we use the Binary Concrete distributions in place of the typically used Gaussian distributions to facilitate the discrete nature of the mask vector. The prior distributions

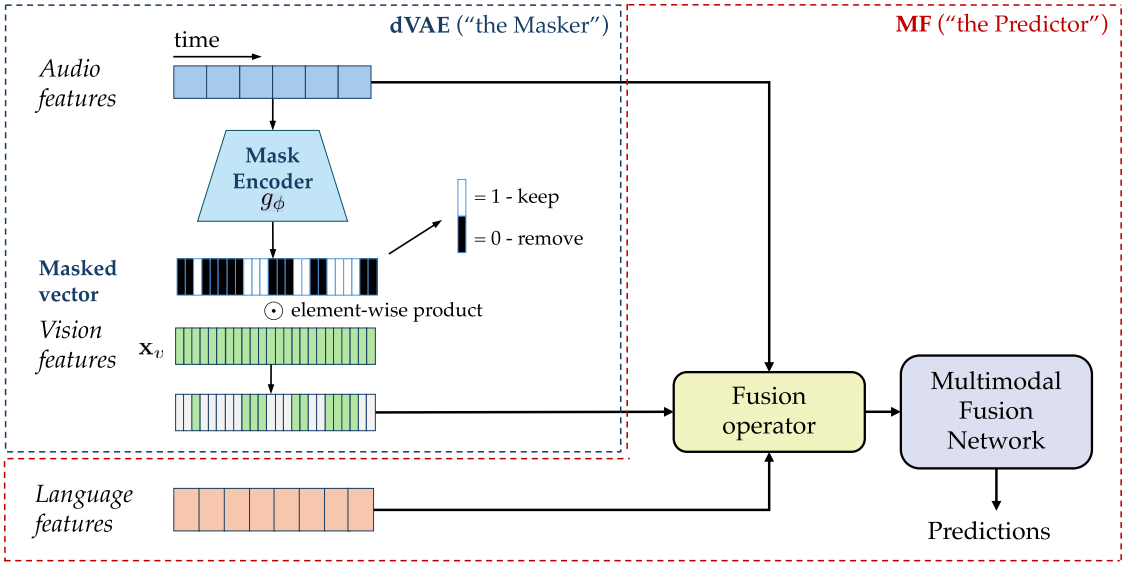


**Fig. 1.** The proposed EMF-dVAE framework.

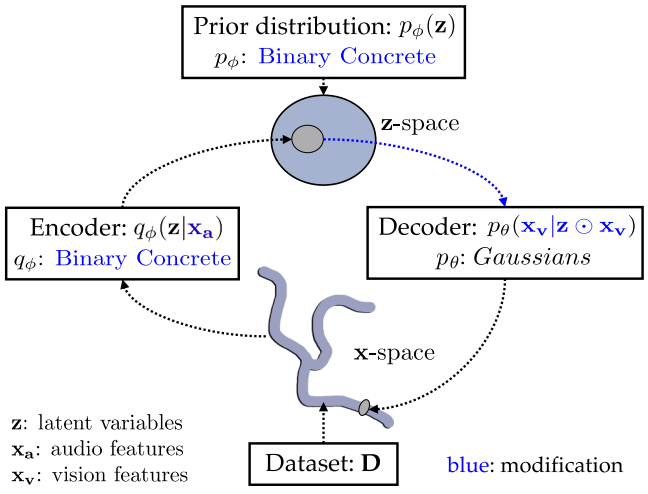


**Fig. 2.** Binary Concrete VAE for unsupervised learning of the masked vector.

can be biased toward a low success rate (high masking ratio) by setting $\pi_{\text{prior}}$ in Eq. (9) to a low value. The posterior distributions can be parameterized with a neural network whose outputs are the $\log \alpha$ values. These modifications allow us to train the Masker model while avoiding the need to check all possible values of $\mathbf{z}$. At inference time, only the encoder of the Masker model is needed for sampling the mask vector. Fig. 3 illustrates the implementation of the dVAE masker model.

#### 3.2.3. Objective function

In the most general case, the training loss function is:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{prediction}} + \lambda(\mathcal{L}_{\text{reconstruction}} + \beta\text{KL-divergence}) \tag{19}$$

where $\lambda$ and $\beta$ are tunable hyperparameters. We selected Mean Squared Error (MSE) as the reconstruction and prediction losses. For $\mathcal{L}_{\text{reconstruction}}$, MSE is appropriate because the embedding space is continuous and geometrically structured, making Euclidean distance a meaningful reconstruction objective. We did not experiment with perceptual loss because the outputs of dVAE are embedding features, and there is no standard pre-trained network that can be used for the downstream evaluation of these features. Contrastive loss was also considered but not adopted, as it is not designed for direct reconstruction of continuous embedding features and requires paired data, which is not available in our setting. For the KL-divergence between Concrete distributions, we used the empirical KL as the estimator.

## 4. Experiments

### 4.1. Dataset

We conducted our experiments on the ETS-Interview dataset, which is a large-scale video corpus of job interviews collected and annotated by [24]. This dataset is chosen to validate our method because of its large size and consistent video durations. The dataset contains 1891 videos, each of which is a 2-minute recording of one of the Mechanical Turk workers located in the United States answering one of eight predefined questions. In total, videos of 260 Tuckers are included in the dataset (as some videos were corrupted, not all Tuckers answered all eight questions). The dataset is equipped with several Likert scale annotations conducted by five independent expert raters, covering personalities and hirability labels. As shown in previous work [9,24,25], the holistic interview performance prediction is the most challenging task; as such, this is the task that we will tackle. Training, validation, and testing sample sizes are 1211, 308, and 372, respectively, with no overlapping participants in the same set. To obtain more samples for

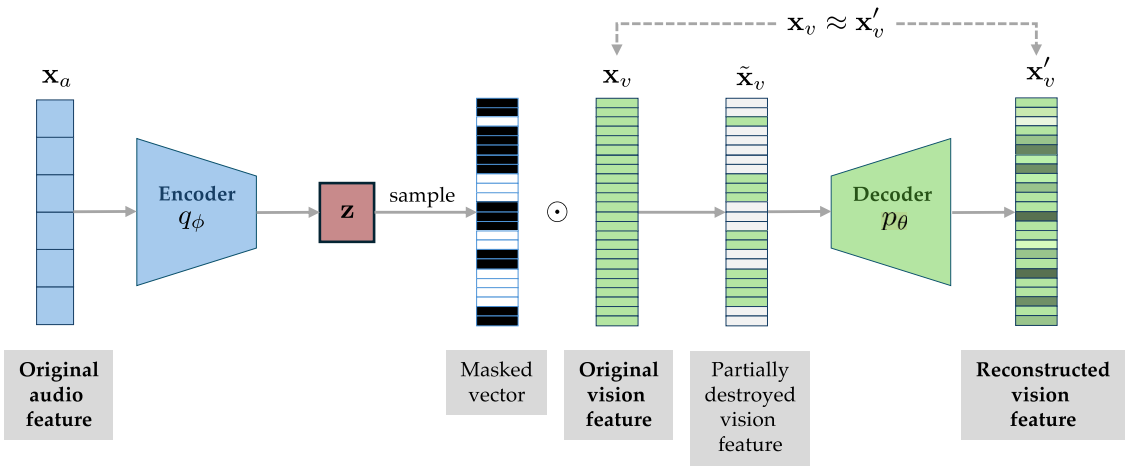


**Fig. 3.** Illustration of discrete VAE for multimodal encoding.

unsupervised training of the Masker, we first remove the background with InSPyReNet[1] [26] and then produce 7 additional augmented versions of the original videos by applying the transformations: vertical flip, horizontal flip, random rotate, gaussian blur, piecewise affine, inverted colors, and multiply (using VidAug[2]). As a result, we obtained 9688 vision-augmented training samples for the unsupervised learning process. We then reduced the frame rate to 16 frames per second, and conducted resizing and central cropping to $224 \times 224$ pixels from the original $640 \times 480$ pixels.

### 4.2. Implementation

#### 4.2.1. Unimodal feature extractors

Directly learning from raw modality forms (e.g., at the pixel level) is ineffective in our case, as a very large amount of data is required to capture meaningful representational embeddings. Instead, we rely on pre-trained unimodal models to extract deep feature representations.

**Vision features.** We extracted vision feature embeddings with VideoMAEv2 [3] [27]. VideoMAEv2 is a very efficient large pre-trained video model, with a masking ratio up to 95%. VideoMAEv2 takes an input of size $16 \times 3 \times 224 \times 224$ (frames × channels × height × width, which is equivalent to a 1-second segment of our video after downsampling, and outputs a 768-dimensional embedding vector. Since the duration of our videos is much longer than a second, we concatenate the segment embedding vectors to get the video embedding tensor of size $T_v \times 768$, where $T_v$ denotes the length of the videos in seconds. Since VideoMAEv2 simultaneously takes 16 frames as its input, it is technically efficient and convenient to map 16 frames to a second. This mapping allows the feature extraction step at inference to be conducted with clear boundaries (for example, extracting features from the 15-second mark to the 18-second mark, as opposed to from the 15.3-second mark to the 18.8-second mark). A finer unit of time can of course be chosen, but there will be an efficiency trade-off, as the number of times the vision encoder model is called will increase.

**Audio features.** We extracted audio feature embeddings with wav2vec 2.0[4] [28]. wav2vec 2.0 is a powerful framework for capturing speech representations directly from waveform input. For each audio sample, two embedding vectors are extracted. The first embedding tensor is the last hidden layer of the model, which we name *audio contextual* features as this embedding captures the contextual representation of the whole audio file (at the cost of losing the temporal dimension). This embedding has the size of $T_{ac} \times 768$. The second embedding tensor is the output of the convolutional component, which is closer to the raw waveform signals and the relative temporal dimension remains intact. This embedding has the size of $T_{aa} \times 512$, and we named it *audio acoustic* features. $T_{aa}$ is much larger than $T_{ac}$ as it is closer to the raw signals and hence keeps more details of the original audio. We opted to make use of the additional *audio acoustic* features because the input of the Masker should have the temporal dimension remain intact, as it (hypothetically) aids the learning of temporal relationships between audio segments and video segments. More details on this are given in Section 4.2.3.

**Language features.** For language features, we extracted the transcripts from the audio files using Whisper[5] and then pass it through the pre-trained RoBERTa-base [29] model. The output is a language embedding tensor of size $T_l \times 768$ for each video, where $T_l$ is the length of transcription.

#### 4.2.2. Base model: the perceiver

As for the base model, we use the same *Perceiver* architecture [30] for the Encoder, the Decoder, and the Predictor. *The Perceiver* was chosen due to its promise of making very few assumptions about the input modality and its ability to handle very long sequences. Unlike the standard attention mechanism in Transformers:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{20}$$

which has a quadratic time and memory complexity due to QKV are in $\mathbb{R}^{M \times D}$ ($M$ is the sequence length and $D$ is the embedding dimension); *the Perceiver* introduced the asymmatric attention by letting $Q \in \mathbb{R}^{N \times D}$ (where $N \ll M$). The resulting cross-attention then has the lower complexity of $\mathcal{O}(MN)$ instead of $\mathcal{O}(M^2)$.

#### 4.2.3. Realization of the framework

Fig. 4 shows the concrete components of the framework. As previously mentioned, we used the *audio acoustic* features as the input to the Mask encoder and the *audio contextual* features as the input to the Predictor. The acoustic features are used for the mask construction as they are closer to the raw waveform and the temporal dimension remains intact, while the contextual features provide a global compression of the whole audio file, making them more suitable for final task prediction. The area bounded by the dotted blue line encapsulates the Masker model, and it can be trained unsupervised. The area bounded by the dotted red line represents the Predictor model, and it can be trained with the label to

[1] https://github.com/nadermx/backgroundremover
[2] https://github.com/okankop/vidaug
[3] Checkpoint: videomaev2-base.
[4] Checkpoint: wav2vec2-base-960h.
[5] Checkpoint: whisper-base.

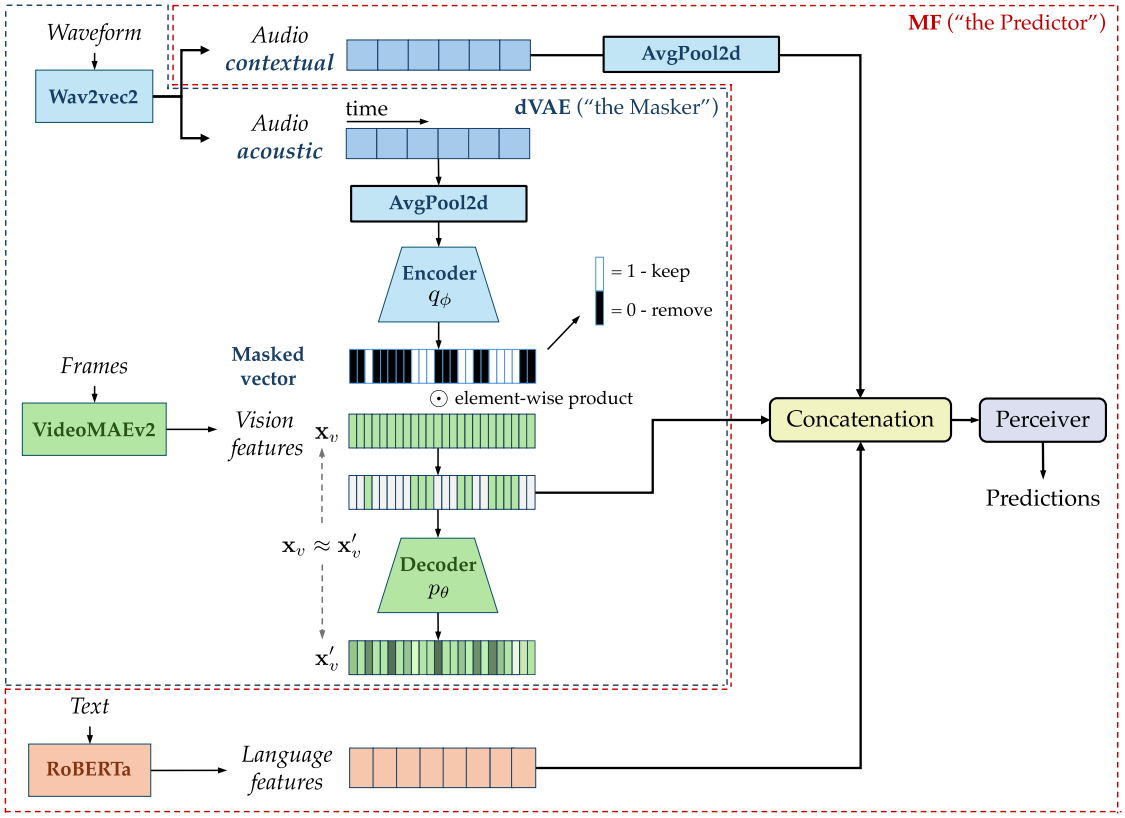


**Fig. 4.** Implementation of the EMF-dVAE framework.

make the final predictions. The unimodal features are simply concatenated and passed to the Perceiver model, where fusion and alignment implicitly take place. We note that the Predictor does not have any requirements regarding its vision input, i.e., it can take either the original vision features or the reconstructed vision features. This design choice makes it easier to evaluate the effectiveness of the Masker. For training, we first train the Masker model with mostly the reconstruction loss, then the KL-divergence term is gradually introduced to penalize the higher masking ratios. Once the Masker model training has converged, we use its output to train the Predictor model.

#### 4.2.4. Hyperparameters and hardware configuration

Table S1 (SM) shows the hyperparameters of the models. All experiments were conducted on a desktop with 1 GPU (Nvidia RTX 4090), 16-core CPU (Intel i9-12900), 32GB of RAM (2 × DDR5), and 1TB of NVMe SSD storage. The training time for the Masker and the Predictor is 24.5 minutes and 3.1 minutes, respectively.

### 4.3. Baseline models and evaluation metrics

For evaluation, we compare our method with three models previously developed for the same dataset (ACII2017 [24], HCII2023 [25], ICMI2024 [9]) and three SOTA models in affective computing (MulT [31], TFR-Net [32], EMT-DLFR [33]). Descriptions of these models are given in Section S3 (SM). For metrics, five commonly employed metrics were used for evaluation, including correlation (Corr), 2-class F1-score (F1), 2-class accuracy (Acc2), 7-class accuracy (Acc7), and mean absolute error (MAE). We keep the metric calculations (such as binary threshold, macro average, etc.) the same as previous work for the ease of comparison.

### 4.4. Main results

**Reconstruction loss vs. KL-divergence.** To train the dVAE masker, we first biased the prior Concrete distributions towards class 0 by setting $\pi$ in Eq. (9) to 0.01. The prior distributions then act as a regularization, as the KL-divergence will pull the posterior distributions towards the prior, i.e., the mask vectors with more zeros are preferred. The KL-divergence weight $\beta$ allows us to control this regularization. Initially, we set $\beta$ to 0 and then gradually increased $\beta$ by 1 after every training epoch. With a small $\beta$, the reconstruction loss will dominate the total VAE loss, allowing the model to reconstruct visual features by simply opening up the mask. As training progresses, the KL-divergence loss will gradually dominate the total loss, forcing the model to reduce the percentage of visible segments at the expense of reconstruction ability. We then save the checkpoints for dVAE when the visible ratio on the validation set reaches 75%, 50%, 25%, 20%, 15%, 10%, and 5%. Training stops at the 5% checkpoint or at a maximum of 100 epochs. Fig. 5 shows the behavior of the reconstruction MAE, KL-divergence, and percentage of visible segments when the temperatures of the posteriors are 0.1, 0.5, or 1. A lower temperature results in greater discrepancy between the probabilities of the binary classes in each mask vector element, but at the cost of higher gradient variance. Conversely, a higher temperature produces softer mask vectors (probabilities closer to 0.5) with lower gradient variance, but at the cost of less discrete masking decisions. With $\tau_{\text{posterior}} = 0.5$, training is smooth and the model converged as expected. With $\tau_{\text{posterior}} = 0.1$, dVAE failed to converge; and with $\tau_{\text{posterior}} = 1$, the model still converges but the interaction between the two losses is more abrupt.

**Sensitivity analysis.** To determine the optimal masking ratio, we conducted experiments on the Predictor using saved dVAE's checkpoints. Fig. 6 shows the 10-run averaged Corr and F1-score of the Predictor when trained with no vision features, all vision features, and vision features produced by the saved checkpoints at masking ratios of 5%,

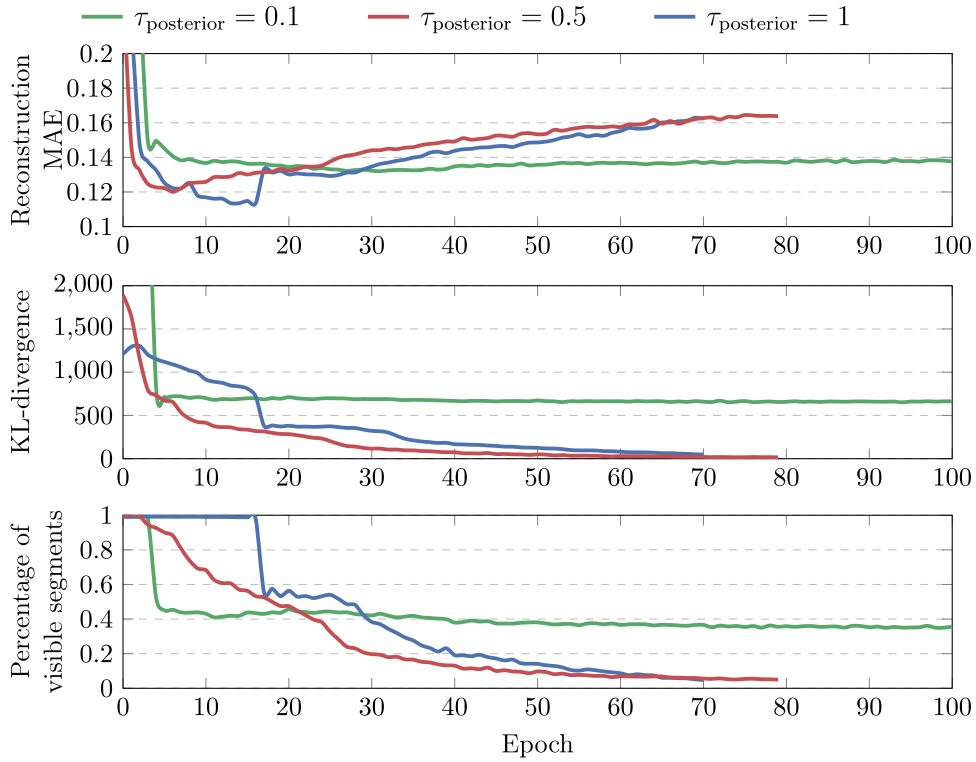


**Fig. 5.** The effects of $\tau_{\text{posterior}}$ on dVAE training stability and convergence.

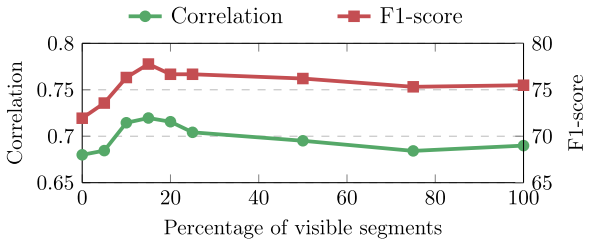


**Fig. 6.** Performance of the Predictor model with different levels of masking ratio produced by dVAE, averaged over 10 seeds.

**Table 1**
Performance results on the ETS-Interview dataset.

| Description | Vision | Corr ↑ | F1 ↑ | Acc2 ↑ | Acc7 ↑ | MAE ↓ |
|---|---|---|---|---|---|---|
| ACII2017 [24] | - | - | 66.38 | 66.40 | - | - |
| HCII2023 [25] | - | - | 70.29 | 70.33 | - | - |
| ICMI2024 [9] | 20% | 0.6792 | 74.73 | 74.57 | 57.53 | 0.5201 |
| MulT [31] | 100% | 0.6174 | 67.83 | 68.1 | 52.24 | 0.5793 |
| TFR-Net [32] | 100% | 0.3860 | 61.95 | 64.07 | 51.79 | 0.6582 |
| EMT-DLFR [33] | 100% | 0.6816 | 72.24 | 72.58 | 55.91 | 0.5259 |
| EMF-dVAE | 15.42% | **0.7196** | **76.64** | **76.77** | **59.28** | **0.4943** |

10%, 15%, 20%, 25%, 50%, and 75%. The performance curve peaks at the 15% masking ratio, suggesting that too few or too many vision features can harm performance.

**Performance results.** Table 1 compares the detailed results of EMF-dVAE at the 15% masking ratio checkpoint with baselines. As the table shows, EMF-dVAE achieves better results across all metrics while using only 15.42% of all available visual features on the test set. In terms of correlation, EMF-dVAE achieves a 0.0404-point improvement over the previous best method on the same dataset (ICMI2024 [9]) and a 0.038-point improvement compared to the SOTA method (EMT-DLFR [33]). In terms of classification, EMF-dVAE also achieves a new high F1-score of 76.64%, a 1.91-point increase compared to ICMI2024 and a 4.4-point increase compared to EMT-DLFR. Out of 10 runs, the best model trained with the dVAE's mask achieves a correlation of 0.7591, while the worst model achieves a correlation of 0.7050. For reference, the minimum and average correlation coefficients of individual human raters' scores to the averaged scores are 0.70 and 0.74 [24]. Our worst and average results therefore (for the first time) reach human-level agreement, and our best model surpasses the average human rater.

**Inference time analysis.** To understand how much time is saved with EMF-dVAE, we constructed two inference pipelines that simulate the deployment environment. The first pipeline (EMF-dVAE) uses the pre-trained encoder of the Masker model to select the salient segments, conducts pre-processing and feature extraction on only the selected segments, and then passes the extracted features to the Predictor model. The second pipeline follows the typical approach, where features are extracted from the whole video and then passed to the predictor models. Three SOTA models are chosen as the predictor models, and there is no Masker model. We ran sequential inference with 10 random videos taken from the validation set and report the averaged statistics in Table 2. On average, EMF-dVAE is able to reduce the total inference time from 52.13 seconds to 18.01 seconds (a 65.5% reduction). In addition to the time saved from the smaller number of vision features to be extracted, the running time of our predictor model is also shorter than that of other methods (0.2 seconds compared to more than 1.31 seconds) due to the smaller number of parameters. We note that the 2.02-second inference time for the Masker model can be decreased in deployment if the model is pre-loaded into GPUs or batch inference is possible.

**Parameter-free masking approaches.** Table 3 compares the performance of EMF-dVAE with two simple parameter-free masking techniques: random masking and uniform masking. Random masking generates binary vectors by thresholding uniformly sampled random values at 0.15. Uniform masking produces the masked binary vector by selecting 15% of segments at evenly-spaced intervals. To our surprise, these two simple masking techniques could achieve relatively high results. That

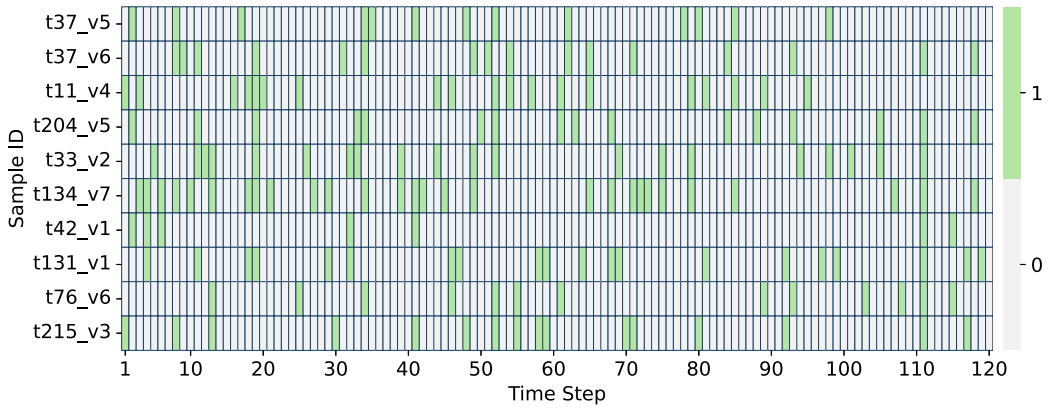


**Fig. 7.** Visualization of the masks produced for 10 random validation samples.

**Table 2**
Inference time (in seconds) of EMF-dVAE compared to typical multimodal fusion approaches (average results over 10 random videos).

| Component | MulT | TFR-Net | EMT-DLFR | EMF-dVAE |
|---|---|---|---|---|
| Masker model | - | - | - | 2.02 |
| Feature extraction | 50.82 | 51.09 | 50.83 | **17.80** |
| Predictor model | 1.31 | 1.71 | 1.32 | **0.20** |
| Total time | 52.13 | 52.80 | 52.15 | **18.01** |

being said, EMF-dVAE outperforms both of these approaches while having smaller standard deviations across all metrics, indicating a more stable and reliable selection of the salient segments.

**Ablation study.** To isolate the contribution of selective visual segments from architectural improvements, we conducted ablation experiments with the following additional settings: the Predictor model with all vision features and without vision features, the Predictor model without audio features, mask learning by Predictor-supervision (the Masker's decoder component is removed), and mask learning with additional language features (EMF-dVAE-L – language is used in addition to audio in the dVAE masker). Table 4 shows that the improved results indeed came from the incorporation of vision modality. EMF-dVAE also achieves better performance than the Predictor with all vision features and features selected by the Predictor-supervised mask learning approach. The last two rows in Table 4 show that the inclusion of language features in the masker model does not significantly improve performance. Possible reasons for this result are the lack of time-series information in the contextual language features and the partial overlap of signals in language and audio, as transcriptions are generated from audio.

**Visualization of the mask.** In Fig. 7, we visualize the masks produced by dVAE on 10 random samples taken from the validation set. The masks are sparse across samples, indicating that there is no posterior collapse (a situation when VAE disregards the input); and the number of masked segments is different between samples: sample `t42_v1` requires only 7 visual segments while sample `t134_v7` requires 28 visual segments. This difference supports our original hypothesis that different videos require different amounts of computation. Fig. 8 shows the histogram of informative segments selected by the model in different videos. Two patterns can be inferred from Fig. 8: (1) there are peaks at the beginning, middle, and end of the videos; and (2) the remaining time steps (excluding the dip at time step 60) are selected with a comparable frequency, which could explain the high performance of the parameter-free uniform masking approach on this dataset. As opposed to selecting only one continuous representative visual segment as in ICMI2024 [9], the figures validate that dVAE can select finer segments across the whole video, as well as a different number of fine-grained segments for different videos, which may contribute to the improved performance.

## 5. Related work

### 5.1. Human-centric video understanding

Human-centric video understanding refers to the realm of tasks designed for machines to learn about human behaviors and emotions. The tasks can range from basic perception recognition (e.g., emotion recognition) from short clips (IEMOCAP [34], MOSEI [35]) to advanced cognitive reasoning (e.g., interview assessment) from long-form videos (ETS-interview [24], MIT-interview [36]). The advanced cognitive reasoning tasks are more challenging due to the computational demand, lack of annotated datasets, and the strong requirement to make use of all available modalities. While the method proposed in this paper was evaluated on an interview assessment dataset, it has the potential to be applied to other advanced cognitive reasoning tasks.

### 5.2. Efficient multimodal deep learning

Dynamic neural networks refer to networks that can adapt their structures or parameters to different inputs during the inference phase [37]. Under this umbrella, our framework can be utilized to build *temporal-wise dynamic models* that adaptively select (remove) frames to be processed. The frame selection problem has been studied intensively with the development of methods such as adaptive frame selection [38,39] and vision token pruning [40–42]. Our work is also related to cross-modal early exiting strategies, such as [43,44], and denoising methods, such as [45,46]. While sharing the general goal and inspirations, our proposed method significantly differs from these related works.

### 5.3. Discrete representation learning

There are many applications in unsupervised learning, language modeling, or reinforcement learning where discrete distributions are encountered. Stochastic neural networks with discrete random variables are developed to provide the theoretical and practical framework for those applications, e.g., the discovery of the Gumbel-Softmax or Concrete distribution [19,20]. In this work, we apply the Binary Concrete distribution for discrete representation learning under the VAE framework. One type of discrete VAE was previously proposed by [47]. EMF-dVAE significantly differs from this approach, as we do not project the approximating posterior and the prior into a continuous space, but future work could compare the two approaches. The Vector Quantised-VAE [48] model was also introduced to produce discrete codes via a codebook. We did not employ this model in EMF-dVAE, since in its standard implementation, the size of the codebook would grow exponentially with respect to the length of the video.

**Table 3**
Comparison with parameter-free approaches.

| Setting | Vision | Corr ↑ | F1 ↑ | Acc2 ↑ | Acc7 ↑ | MAE ↓ |
|---|---|---|---|---|---|---|
| Random masking | 15% | 0.7049 (± 0.021) | 75.43 (± 2.3) | 75.62 (± 2.4) | 58.28 (± 2.9) | 0.5064 (± 0.02) |
| Uniform masking | 15% | 0.7117 (± 0.020) | 75.86 (± 3.4) | 76.02 (± 3.2) | 56.56 (± 4.3) | 0.5196 (± 0.03) |
| EMF-dVAE | 15.42% (± 0.16) | **0.7196** (± 0.016) | **76.64** (± 1.96) | **76.77** (± 1.96) | **59.28** (± 2.23) | **0.4943** (± 0.016) |

**Table 4**
Ablation study results.

| Component | Vision | Corr ↑ | F1 ↑ | Acc2 ↑ | Acc7 ↑ | MAE ↓ |
|---|---|---|---|---|---|---|
| Predictor only | 0% | 0.6816 | 72.95 | 73.44 | 57.12 | 0.5270 |
| Predictor only | 100% | 0.6819 | 74.25 | 74.44 | 55.97 | 0.5256 |
| Predictor w/o audio | 0% | 0.6452 | 70.40 | 70.67 | 52.12 | 0.5621 |
| Predictor w/o audio | 100% | 0.6469 | 74.63 | 74.82 | 55.76 | 0.5449 |
| Predictor-supervised mask learning | 16.96% | 0.6561 | 72.57 | 72.82 | 54.83 | 0.5555 |
| EMF-dVAE-L | 14.16% | 0.7148 | **76.80** | **76.91** | 58.95 | 0.4989 |
| EMF-dVAE | 15.42% | **0.7196** | 76.64 | 76.77 | **59.28** | **0.4943** |

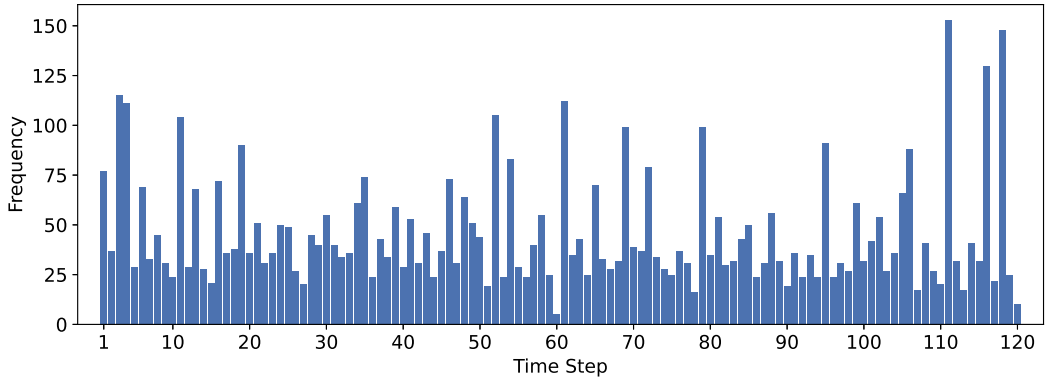


**Fig. 8.** Histogram of informative segments selected by the dVAE across different videos. The x-axis is the timestamp of the segments within the videos, and the y-axis is the number of times each segment is selected across the total set of 308 validation videos.

## 6. Conclusion

This work introduces EMF-dVAE, an efficient multimodal fusion framework for human-centric video understanding. EMF-dVAE uses only 15.42 percent of the vision features yet achieves state-of-the-art performance, while reducing the inference time from 52.13 seconds to 18.01 seconds for 2-minute videos. However, several open questions remain. First, what is the generalization ability of EMF-dVAE? In this work, we conducted experiments on only one large-scale interview video dataset. It would be of interest to investigate how EMF-dVAE performs across a wider range of tasks and datasets. Second, while dVAE relies on the Binary Concrete distribution to produce the mask vector, alternative approaches exist. Future work could explore those approaches and compare the pros and cons of each. Third, we did not employ audio augmentation methods in this work. The incorporation of those methods could potentially improve the Masker's ability to learn robust audio-visual correlations. Finally, EMF-dVAE was evaluated on an English-speaking dataset. For languages with significantly different intonation or word order, effective knowledge fusion remains an open challenge.

## CRediT authorship contribution statement

**Hung Le:** Writing – review & editing, Writing – original draft, Visualization, Methodology, Investigation, Conceptualization; **Hung-Hsuan Huang:** Writing – review & editing, Methodology, Conceptualization; **Candy Olivia Mawalim:** Writing – review & editing, Methodology; **Chee Wee Leong:** Writing – review & editing, Methodology, Data curation; **Shogo Okada:** Writing – review & editing, Supervision, Methodology, Funding acquisition, Conceptualization.

## Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

## Acknowledgment

This work was partially supported by JSPS KAKENHI (26H00000), and JST CREST, Japan (JPMJCR2563), and JST CRONOS (JPMJCS24K7).

## Data availability

The data that has been used is confidential.

## Supplementary material

Supplementary material associated with this article can be found, in the online version, at 10.1016/j.inffus.2026.104613

## References

[1] T. Baltrusaitis, C. Ahuja, L.-P. Morency, Multimodal machine learning: a survey and taxonomy, IEEE Trans. Pattern Anal. Mach. Intell. 41 (2) (2019) 423-443. https://doi.org/10.1109/TPAMI.2018.2798607

[2] C. Wu, J. Liang, L. Ji, F. Yang, Y. Fang, D. Jiang, N. Duan, NÜWA: Visual Synthesis Pre-training for Neural visUal World creAtion, 2022, pp. 720–736. https://doi.org/10.1007/978-3-031-19787-1_41

[3] S. Huang, L. Dong, W. Wang, Y. Hao, S. Singhal, S. Ma, T. Lv, L. Cui, O.K. Mohammed, B. Patra, Q. Liu, K. Aggarwal, Z. Chi, J. Bjorck, V. Chaudhary, S. Som, X. Song, F. Wei, Language is not all you need: aligning perception with language models, in: Proceedings of the 37th International Conference on Neural Information Processing Systems, NIPS '23, Curran Associates Inc., Red Hook, NY, USA, 2023.

[4] Y. He, R. Cheng, G. Balasubramaniam, Y.-H.H. Tsai, H. Zhao, Efficient modality selection in multimodal learning, J. Mach. Learn. Res. 25 (47) (2024) 1–39.

[5] Z. Zhang, P. Pham, W. Zhao, K. Wan, Y.-J. Li, J. Zhou, D. Miranda, A. Kale, C. Xu, Treat Visual Tokens as Text? But Your MLLM Only Needs Fewer Efforts to See, 2024. arXiv:2410.06169[cs.CV]

[6] Z. Xu, K.D. Nguyen, P. Mukherjee, S. Bagchi, S. Chaterji, Y. Liang, Y. Li, Learning to Inference Adaptively for Multimodal Large Language Models, 2025, arXiv:2503.10905[cs.AI]

[7] P.V. Dantas, W. Sabino da Silva, L.C. Cordeiro, C.B. Carvalho, A comprehensive review of model compression techniques in machine learning, Appl. Intell. 54 (22) (2024) 11804-11844. https://doi.org/10.1007/s10489-024-05747-w

[8] Z. Wu, C. Xiong, Y.-G. Jiang, L.S. Davis, LiteEval: a coarse-to-fine framework for resource efficient video recognition, Curran Associates Inc., Red Hook, NY, USA, 2019.

[9] H. Le, S. Li, C.O. Mawalim, H.-H. Huang, C.W. Leong, S. Okada, Do we need to watch it all? efficient job interview video processing with differentiable masking, in: Proceedings of the 26th International Conference on Multimodal Interaction, ICMI '24, Association for Computing Machinery, New York, NY, USA, 2024, p. 565-574. https://doi.org/10.1145/3678957.3685718

[10] A. Zadeh, M. Chen, S. Poria, E. Cambria, L.-P. Morency, Tensor Fusion Network for Multimodal Sentiment Analysis, 2017, arXiv:1707.07250[cs.CL]

[11] W. Wang, D. Tran, M. Feiszli, What makes training multi-modal classification networks hard?, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2020, pp. 12695–12705.

[12] X. Sun, R. Panda, C.-F.R. Chen, A. Oliva, R. Feris, K. Saenko, Dynamic network quantization for efficient video inference, in: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2021, pp. 7375–7385.

[13] R. Gao, T.-H. Oh, K. Grauman, L. Torresani, Listen to look: action recognition by previewing audio, in: 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2020, pp. 10454–10464. https://doi.org/10.1109/CVPR42600.2020.01047

[14] J. Schulman, N. Heess, T. Weber, P. Abbeel, Gradient estimation using stochastic computation graphs, in: Proceedings of the 29th International Conference on Neural Information Processing Systems - Volume 2, NIPS'15, MIT Press, Cambridge, MA, USA, 2015, p. 3528-3536.

[15] A. Mnih, D.J. Rezende, et al., Variational inference for Monte Carlo objectives, in: Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48, ICML'16, JMLR.org, 2016, p. 2188-2196.

[16] G. Tucker, A. Mnih, C.J. Maddison, D. Lawson, J. Sohl-Dickstein, et al., REBAR: Low-variance, unbiased gradient estimates for discrete latent variable models, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17, Curran Associates Inc., Red Hook, NY, USA, 2017, p. 2624-2633.

[17] D.P. Kingma, M. Welling, Auto-encoding variational bayes, in: Y. Bengio, Y. LeCun (Eds.), 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings, 2014.

[18] E.J. Gumbel, Statistical Theory of Extreme Values and Some Practical Applications: A Series of Lectures, 33 of National Bureau of Standards Applied Mathematics Series, U.S. Department of Commerce, Washington, D.C., 1954.

[19] E. Jang, S. Gu, B. Poole, Categorical reparameterization with gumbel-Softmax, in: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings, OpenReview.net, 2017.

[20] C.J. Maddison, A. Mnih, Y.W. Teh, The concrete distribution: a continuous relaxation of discrete random variables, in: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings, OpenReview.net, 2017.

[21] D.P. Kingma, M. Welling, An introduction to variational autoencoders, Found. Trends Mach. Learn. 12 (4) (2019) 307-392. https://doi.org/10.1561/2200000056

[22] T. Baltrusaitis, A. Zadeh, Y.C. Lim, L.-P. Morency, Openface 2.0: facial behavior analysis toolkit, in: 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018), 2018, pp. 59–66. https://doi.org/10.1109/FG.2018.00019

[23] X. Zhang, J. Yoon, M. Bansal, H. Yao, Multimodal representation learning by alternating unimodal adaptation, in: 2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2024, pp. 27446–27456. https://doi.org/10.1109/CVPR52733.2024.02592

[24] L. Chen, R. Zhao, C.W. Leong, B. Lehman, G. Feng, M.E. Hoque, Automated video interview judgment on a large-sized corpus collected online, in: 2017 Seventh International Conference on Affective Computing and Intelligent Interaction (ACII), 2017, pp. 504–509. ISSN: 2156–8111. https://doi.org/10.1109/ACII.2017.8273646

[25] H. Le, S. Li, C.O. Mawalim, H.-H. Huang, C.W. Leong, S. Okada, Investigating the effect of linguistic features on personality and job performance predictions, in: A. Coman, S. Vasilache (Eds.), Social Computing and Social Media, Lecture Notes in Computer Science, Springer Nature Switzerland, Cham, 2023, pp. 370–383. https://doi.org/10.1007/978-3-031-35915-6

[26] T. Kim, K. Kim, J. Lee, D. Cha, J. Lee, D. Kim, Revisiting image pyramid structure for high resolution salient object detection, in: Proceedings of the Asian Conference on Computer Vision, 2022, pp. 108–124.

[27] L. Wang, B. Huang, Z. Zhao, Z. Tong, Y. He, Y. Wang, Y. Wang, Y. Qiao, VideoMAE V2: scaling video masked autoencoders with dual masking, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023, pp. 14549–14560.

[28] A. Baevski, H. Zhou, A. Mohamed, M. Auli, Wav2vec 2.0: a framework for self-supervised learning of speech representations, in: Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS '20, Curran Associates Inc., Red Hook, NY, USA, 2020.

[29] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, V. Stoyanov, et al., RoBERTa: A Robustly Optimized BERT Pretraining Approach, 2019, arXiv:1907.11692[cs]. https://doi.org/10.48550/arXiv.1907.11692

[30] A. Jaegle, F. Gimeno, A. Brock, A. Zisserman, O. Vinyals, J. Carreira, Perceiver: general perception with iterative attention, Corr abs/2103.03206 (2021). 2103.03206

[31] Y.-H.H. Tsai, S. Bai, P.P. Liang, J.Z. Kolter, L.-P. Morency, R. Salakhutdinov, Multimodal transformer for unaligned multimodal language sequences, in: A. Korhonen, D. Traum, L. Màrquez (Eds.), Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, Association for Computational Linguistics, Florence, Italy, 2019, pp. 6558–6569. https://doi.org/10.18653/v1/P19-1656

[32] Z. Yuan, W. Li, H. Xu, W. Yu, Transformer-based feature reconstruction network for robust multimodal sentiment analysis, in: Proceedings of the 29th ACM International Conference on Multimedia, MM '21, Association for Computing Machinery, New York, NY, USA, 2021, p. 4400-4407. https://doi.org/10.1145/3474085.3475585

[33] L. Sun, Z. Lian, B. Liu, J. Tao, Efficient multimodal transformer with dual-Level feature restoration for robust multimodal sentiment analysis, IEEE Trans. Affect. Comput. 15 (1) (2024) 309-325. https://doi.org/10.1109/TAFFC.2023.3274829

[34] C. Busso, M. Bulut, C.-C. Lee, A. Kazemzadeh, E. Mower, S. Kim, J.N. Chang, S. Lee, S.S. Narayanan, IEMOCAP: Interactive emotional dyadic motion capture database, Lang. Resour. Eval. 42 (4) (2008) 335–359. https://doi.org/10.1007/s10579-008-9076-6

[35] A. Bagher Zadeh, P.P. Liang, S. Poria, E. Cambria, L.-P. Morency, Multimodal language analysis in the wild: CMU-MOSEI dataset and interpretable dynamic fusion graph, in: I. Gurevych, Y. Miyao (Eds.), Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Association for Computational Linguistics, Melbourne, Australia, 2018, pp. 2236–2246. https://doi.org/10.18653/v1/P18-1208

[36] I. Naim, M.I. Tanveer, D. Gildea, M.E. Hoque, Automated prediction and analysis of job interview performance: the role of what you say and how you say it, in: 2015 11th IEEE International Conference and Workshops on Automatic Face and Gesture Recognition (FG), 1, 2015, pp. 1–6. https://doi.org/10.1109/FG.2015.7163127

[37] Y. Han, G. Huang, S. Song, L. Yang, H. Wang, Y. Wang, Dynamic neural networks: a survey, IEEE Trans. Pattern Anal. Mach. Intell. 44 (11) (2022) 7436–7456. https://doi.org/10.1109/TPAMI.2021.3117837

[38] Z. Wu, C. Xiong, C.-Y. Ma, R. Socher, L.S. Davis, AdaFrame: Adaptive Frame Selection for Fast Video Recognition, 2019, arXiv:1811.12432[cs.CV]

[39] H. Tao, Q. Duan, An adaptive frame selection network with enhanced dilated convolution for video smoke recognition, Expert Syst. Appl. 215 (2023) 119371. https://doi.org/10.1016/j.eswa.2022.119371

[40] Z. Kong, P. Dong, X. Ma, X. Meng, W. Niu, M. Sun, X. Shen, G. Yuan, B. Ren, H. Tang, M. Qin, Y. Wang, SPViT: enabling faster vision transformers via latency-Aware soft token pruning, in: S. Avidan, G. Brostow, M. Cissé, G.M. Farinella, T. Hassner (Eds.), Computer Vision – ECCV 2022, Springer Nature Switzerland, Cham, 2022, pp. 620–640.

[41] S. Wei, T. Ye, S. Zhang, Y. Tang, J. Liang, Joint token pruning and squeezing towards more aggressive compression of vision transformers, in: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2023, pp. 2092–2101.

[42] S. Zhang, Q. Fang, Z. Yang, Y. Feng, LLaVA-Mini: Efficient Image and Video Large Multimodal Models with One Vision Token, 2025, arXiv:2501.03895[cs.CV]

[43] C. Sun, M. Chen, C. Zhu, S. Zhang, P. Lu, J. Chen, Listen with seeing: cross-Modal contrastive learning for audio-visual event localization, IEEE Trans. Multimed. 27 (2025) 2650–2665. https://doi.org/10.1109/TMM.2025.3535359

[44] Q. Wu, W. Lin, Y. Zhou, W. Ye, Z. Zeng, X. Sun, R. Ji, Accelerating multimodal large language models via dynamic visual-Token exit and the empirical findings, in: D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, N. Chen (Eds.), Advances in Neural Information Processing Systems, 38, Curran Associates, Inc., 2025, pp. 168378–168403.

[45] W. Zhou, B. Tang, R. Cong, Q. Jiang, Turbidity-similarity decoupling: feature-Consistent mutual learning for underwater salient object detection, IEEE Trans. Image Process. 35 (2026) 495–510. https://doi.org/10.1109/TIP.2025.3648880

[46] W. Zhou, J. Xie, C. Xu, Semantic prompt and graph-Convolution-Structure distillation framework for semantic segmentation of remote sensing images, IEEE Trans. Neural Netw. Learn. Syst. (2026) 1–14. https://doi.org/10.1109/TNNLS.2026.3675381

[47] J.T. Rolfe, Discrete Variational Autoencoders, 2017, arXiv:1609.02200[stat.ML]

[48] A. van den Oord, O. Vinyals, K. Kavukcuoglu, Neural discrete representation learning, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, NIPS'17, Curran Associates Inc., Red Hook, NY, USA, 2017, p. 6309–6318.