

Fine-grained Multi-Modal Self-Supervised Learning

Duo Wang*
wd263@cam.ac.uk

Department of Computer Science and
Technology,
University of Cambridge,
Cambridge, UK
Huawei R&D UK,
302 Cambridge Science Park,
Cambridge, UK

Salah Karout
Salah.Karout@huawei.com

Abstract

Multi-Modal Self-Supervised Learning from videos has been shown to improve model’s performance on various downstream tasks. However, such Self-Supervised pre-training requires large batch sizes and a large amount of computation resources due to the noise present in the uncurated data. This is partly due to the fact that the prevalent training scheme is trained on coarse-grained setting, in which vectors representing the whole video clips or natural language sentences are used for computing similarity. Such scheme makes training noisy as part of the video clips can be totally not correlated with the other-modality input such as text description. In this paper, we propose a fine-grained multi-modal self-supervised training scheme that computes the similarity between embeddings at finer-scale (such as individual feature map embeddings and embeddings of phrases), and uses attention mechanisms to reduce noisy pairs’ weighting in the loss function. We show that with the proposed pre-training scheme, we can train smaller models, with smaller batch-size and much less computational resources to achieve downstream tasks performances comparable to State-Of-The-Art, for tasks including action recognition and text-image retrievals.

1 Introduction

Self-Supervised Learning has recently emerged as an effective method for learning good data representations from unlabelled data to improve downstream task performances [1, 2, 14, 25, 30]. Among these methods, Multi-Modal Self-Supervised Learning from uncurated videos are particularly effective, achieving downstream task performances comparable to Supervised Pre-Training for tasks such as action recognition [1, 25, 30], information retrieval [30] and video question answering [9]. These methods mostly use Noise Contrastive Estimation [15] to exploit the inherent correlations between different modalities in multi-modal video data. For example, in the case of video-text learning, embeddings of video frames and narrations belonging to the same video clip are pushed closer to each other while embeddings belonging to different video clips are pushed further apart.

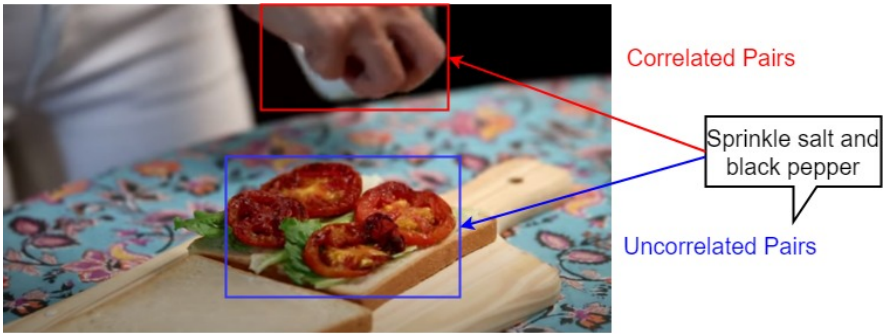


Figure 1: Illustration of uncorrelated pairing between video content and accompanied narrations. Example is from YouCookII [43] dataset.

These approaches, while effective in boosting downstream task performances, make simplifying assumptions about the multi-modal data. These methods embed a whole video clip or a whole natural language sentence into a single embedding vector. However, for uncensored videos, narrations are mostly only correlated with a small part of the video clip. For example, in Figure 1, the narration “Sprinkle salt and black pepper” does not refer to other parts of the scene such as tomatoes, lettuces and bread. Computing contrastive loss between embedding vectors of the whole video clip and narration will push embeddings of unrelated parts (e.g. “tomato” in the video and the word “salt” in the narration) closer. To reduce such noise, contrastive learning approaches [9, 19, 25] usually requires large batch sizes containing many negative pairs to pull embedding of unrelated parts further, and consequently requires a large amount of computational resources for training. In addition, these methods often use large models for boosting performance, and thus neglect terminal device application scenarios (e.g. smartphone), where computation resources are very limited.

In this paper we propose a Fine-Grained Multi-Modal Self-Supervised Learning (FG-MMSSL) method, which computes contrastive loss at finer granularity level, such as between embeddings of spatial-temporal regions in video modality and phrases in text modality. In order to reduce the loss variance caused by the uncorrelated finer-granularity embedding pairs, we multiply the computed similarities between each pair with an estimated importance score to scale down the ratio of noise-inducing pairs’ contribution in the loss function. These importance scores are computed using an attention mechanism which matches query and key embeddings generated from finer-granularity embeddings of different modalities. Through our experiments, we show that our FG-MMSSL training scheme can train smaller models with smaller batch sizes on platforms with a smaller amount of computational resources, and achieve downstream task performances comparable to state-of-the-art SSL pretraining methods (for action classification and text-to-video retrieval), or surpass supervised pretraining methods (for temporal action localisation).

2 Related Works

Single-Modality Self-Supervised Learning: Single-Modality Self supervised learning usually uses pretext tasks to automatically generate differentiable learning signals from the data itself in order to train the feature extraction neural networks. For the case of image

modality, such tasks include predicting artificial rotations [13], colourisation [41, 42] and feature clustering [9]. Recently Contrastive Learning [16] has become increasingly popular for learning both visual [9, 19], audio [6, 28] and natural language [10] representations. The method is to push positive pairs' embedding closer while pulling negative pairs' embedding further apart. Positive pairs are usually created by applying data augmentations on the same data point to create pairs that share contents but not lower-level data statistics.

Methods are also developed for video modality self-supervised learning using different pre-text tasks. Such pre-text tasks include future prediction [17], sequence order prediction [26], playback speed [9, 56] and spatial transformations [20]. Recently there are also works [18] that focus on fewer data SSL pre-training by mining hard positives using proximity in optical flow modality's embeddings.

Multi-Modality Self-Supervised Video Representation Learning: Video data usually has multiple modalities, including visual, audio and text. The time synchronised nature of these modalities provides a natural way of extracting positive pairs co-located in time without the need of pre-text tasks. Many methods are developed for audio-visual learning [9, 6, 29] where co-occurrence of action and sound serve as learning signals. Methods have also been developed for speech-visual learning [10, 9, 25], which use the recently published HowTo100M instructional video dataset [24] to exploit the correlation between spoken instructions and actions performed in the videos. These self-supervised methods achieved downstream task performance on-par with supervised SOTA results for tasks such as action recognition and text-image retrieval. There are also concurrent works [8, 27] introducing mechanisms to scale a pair's contribution to the loss function. Among them, Morgado et al [27] use cumulative distributions of the moving averages of dot-product similarities, while Chen et al [8] uses pseudo-masks generated with smoothed Heaviside function. These methods differ from our work in that they are not using a separate attention mechanism to compute the scaling weights, which is shown to be effective in this paper.

Attention: Attention Mechanism, first popularised in the Transformer architecture [54], has now seen wide applications in different areas including Natural Language Processing [52, 54], Image Classification [10], Speech Recognition [6, 58] and graph data processing [55]. While Attention mechanism has also been applied for video tasks such as video retrieval [12]. While attention mechanism has been used in Multi-Modal Self-Supervised Learning for processing within each modality [14, 60], to our best knowledge, our work is the first to explore using attention mechanism in cross modality contrastive loss computation.

3 Method

3.1 Background: Multi-Modal Contrastive Learning

Self-supervised learning aims to learn a function, usually parameterised by a neural network f , that can produce general-purpose feature representation $z = f(x)$ for data x . By training on a large unlabelled dataset, the learnt function f can be then transferred to downstream task to boost performance over training from scratch. The most popular method for SSL is noise contrastive learning [9, 19], which we will briefly review in this section.

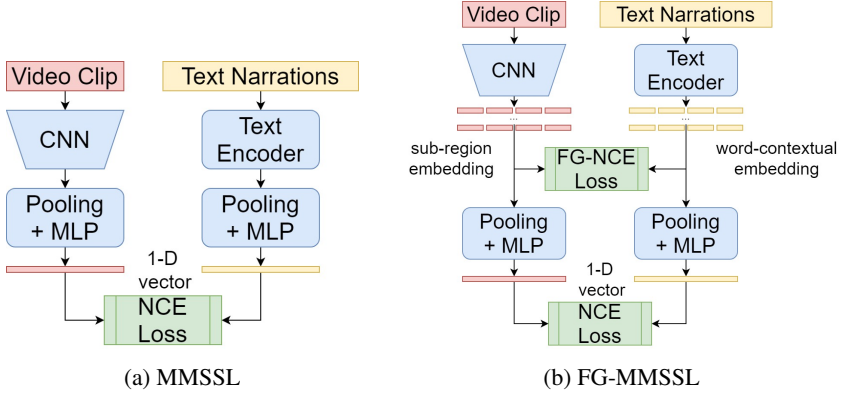


Figure 2: Comparison of (a) Multi-Modal Self-Supervised Learning (MMSSL) and (b) our Fine-grained Multi-Modal Self-Supervised Learning (FG-MMSSL) approach.

The central idea of contrastive learning is that embeddings of inputs containing the same (or similar) contents (positive pairs) are pushed closer together in the embedding space while embeddings of input containing different contents (negative pairs) are pushed further apart. While Single-Modality contrastive learning [9, 19] usually uses different data augmentations of the same data point to generate positive pairs that share the content but not the lower-level statistics, in multi-modal data, such positive pairs naturally exist, such as video clips and accompanying narrations. For two modality video-text case, input video clip x and narration text y are first embedded by a modality-specific embedding module $f_m(x)$, where $m \in \{1, 2\}$ denotes the modality. Then the embeddings are used for computing the multi-modal Noise Contrastive Estimation (NCE) [28] loss as:

$$-\sum_{i=1}^N \log \left(\frac{\sum_{x,y \in \mathcal{P}_i} e^{f_1(x)^T f_2(y)}}{\sum_{x,y \in \mathcal{P}_i} e^{f_1(x)^T f_2(y)} + \sum_{x',y' \in \mathcal{N}_i} e^{f_1(x')^T f_2(y')}} \right) \quad (1)$$

Here, \mathcal{P} denotes positive pairs while \mathcal{N} denotes negative pairs. Multi-Modal Contrastive approaches [9, 8, 25] usually sample positive pairs as video clips and the text narrations that are nearest in the time domain from the same video source, and sample negative pairs as video clips and text narrations from different video sources.

3.2 Motivation for Fine-Grained MMSSL

In equation 1, a whole video clip x and text narrations y containing multiple sentences are both embedded into a 1-dimensional feature vector by f_1 and f_2 . Minimising the Contrastive Loss pushes x and y sampled from positive pairs \mathcal{P}_i closer. However, many spatial temporal sub-parts of the video clips are actually not correlated with phrase sub-parts in the text narrations, as illustrated in Figure 1. Therefore computing similarities between the coarsely embedded feature vectors is inherently noisy as the coarsely embedded vectors are pooled from features of these sub-parts. In this section, we perform a detailed analysis on this subject.

In most MMSSL approaches [9, 25], f_m consists of a Feature Extractor (Convolutional Neural Network for video modality; Multi-Layer Perception or Transformer for text modality), a

pooling layer and a projection layer. For the video modality, input video clip x are passed through a 3D CNN, which output a final feature map H , which are arrays of feature vectors h_i where i index over the flattened feature maps' height, width and time dimension. The feature map is then pooled by a pooling function $Pool$ to generate a single 1-dimensional feature vector. Here we use the most widely used Average Pooling $AvgPool(h_i) = \frac{1}{N} \sum_{i \in N} h_i$ in our analysis. The final projection layer is a linear layer $W \times AvgPool(h_i) + b$ that projects to a common feature space shared with other modalities. We can then rewrite the dot-product terms in equation 1 as:

$$e^{f_1(x)^T f_2(y)} = e^{(W \frac{1}{N} \sum_{i \in N} h_i + b)^T f_2(y)} = e^{\frac{1}{N} \sum_{i \in N} h_i^T W^T f_2(y) + b^T f_2(y)} \quad (2)$$

Where the $h_i^T W^T f_2(y)$ computes similarities between each feature map vectors h_i with the text embedding $f_2(y)$. For the dot-product term in the numerator of Equation 1, minimising the Contrastive Loss maximises the dot-product term, which subsequently maximises all $h_i^T W^T f_2(y)$ pairs for $i \in N$, even though many feature map vectors h_i are uncorrelated with text embedding $f_2(y)$. Maximising these uncorrelated pairs introduce noise into the learning signal, thereby making the training longer and requiring larger batch size to even out sampling discrepancies. While we only perform the analysis on the video modality, similar analysis can be done for other modalities. Please refer to Appendix A for a detailed discussion.

3.3 Fine-Grained MMSSL with noise suppression

In order to reduce noise, ideally we want to minimise the contribution of the uncorrelated positive sub-part pairs to the loss function, and maximise the contribution of the uncorrelated negative sub-part pairs. However in the self-supervised learning setting there is no label information to identify these pairs. Here we propose FG-MMSSL, a framework that computes an additional Contrastive Loss computed directly using the sub-part pairs, and utilise a cross-modal attention mechanism to generate attention variables which scales the contributions from uncorrelated pairs. The additional loss \mathcal{L}_{fg} is computed as:

$$-\mathcal{L}_{fg} = \sum_{i=1}^N \log \left(\frac{\sum_{\mathbf{h}, \mathbf{l} \in \mathcal{P}_i} \sum_{i,j} e^{a_{i,j} f_1(h_i)^T f_2(l_j)}}{\sum_{\mathbf{h}, \mathbf{l} \in \mathcal{P}_i} \sum_{i,j} e^{a_{i,j} f_1(h_i)^T f_2(l_j)} + \sum_{\mathbf{h}', \mathbf{l}' \in \mathcal{N}_i} \sum_{i,j} e^{\frac{1}{1+a_{i,j}} f_1(h'_i)^T f_2(l'_j)}} \right) \quad (3)$$

Here \mathbf{h} and \mathbf{l} are intermediate dense feature arrays extracted from the neural network pipeline of each modality. Specifically \mathbf{h} is the feature map output from the last convolutional layer while \mathbf{l} is the array of hidden representations (otherwise known as contextualised word representations) from text encoders. h_i denotes each grid location in the feature map \mathbf{h} while l_i denotes individual feature vectors in \mathbf{l} . $a_{i,j}$ is an attention variable which scales the dot product of sub-part pair (h_i, l_j) . In positive pairs, we directly multiply $a_{i,j}$ with the dot-product of positive embedding pairs such that more correlated pairs are given a larger weighting. We multiply $\frac{1}{1+a_{i,j}}$ with negative embedding pairs such that uncorrelated pairs are given a larger weighting. $a_{i,j}$ is computed as:

$$a_{i,j} = \text{softmax}_i \left(\frac{k_i^T q_j}{\sqrt{d_k}} \right); \quad k_i = \text{Linear}(h_i); \quad q_j = \text{Linear}(l_j) \quad (4)$$

Here k_i is the key vector generated by applying a linear layer on feature maps h_i of the video modality, while q_j is the query vector generated by applying a linear layer on the array of

hidden representations l_i from text modality. Attention parameter $a_{i,j}$ is then computed using the standard attention mechanism [24], as shown in equation 4. We choose to generate queries from text-modality and keys from video-modality. This is because as narration texts are describing the videos, most of the text are correlated with parts of the video. However the reverse is not true, as videos contain many parts (such as background) that are not described in the text. Thus for each query generated from text embedding, there is a higher chance that there will be a matching key in video embedding. It is tempting to think that multiplying embedding’s dot-product similarity with the attention variable, which is essentially another dot-product, is the same as square the dot-product term as $(f_1(h_i)^T f_2(l_j))^2$. However, in our ablation study, we show that having a separate attention mechanism is essential for improving performance.

3.4 Training Details

For self-supervised pre-training, the model is trained with the following loss function:

$$\mathcal{L} = \mathcal{L}_{cg} + \beta \mathcal{L}_{fg} + \gamma \mathcal{L}_{reg} \quad (5)$$

Here \mathcal{L}_{cg} is the typical NCE loss function computed between whole video-clip and text embeddings. For this loss, we use MIL-NCE [25] loss, a Multi-Instance NCE loss function. \mathcal{L}_{fg} is the fine-grained loss as described in Equation 4. We use the same hyper-parameter setting as in [25]. β is a factor that balances between \mathcal{L}_{cg} and \mathcal{L}_{fg} . Through hyper-parameter search, we set $\beta = 0.001$ as this gives the best result. The optimal value of β is small due to the fact that \mathcal{L}_{fg} is on a larger scale than \mathcal{L}_{cg} . \mathcal{L}_{reg} is an L2-Regularisation loss applied on model parameters. We set γ , the coefficient of the regularisation term, as $1e^{-7}$. To train our model, we use ADAM [26] Optimiser with initial the learning rate of $1e^{-3}$ with linear warm up of 5k steps. We decay the learning rate twice by the factor of 10 at epoch 100 and 200. For self-supervised pre-training, we train for in total 300 epochs.

4 Evaluation

We first describe implementations details of our methods in section 4.1. To show the generality of the learnt representation, we then present downstream task performance evaluation for three different tasks, which are Action Recognition in Section 4.2, text-to-video retrieval in Section 4.3, and temporal action detection in Section 4.4. Finally we perform ablation studies to investigate the effectiveness of various design choices of our method.

4.1 Implementation Details

Pre-training dataset: For self-supervised pre-training, we use the large-scale instructional video dataset named HowTo100M [24]. This dataset contains instructional videos with text narrations generated with automatic speech recognition. We follow the data processing procedure used in MIL-NCE [25]. Briefly we randomly sample fixed length clips of 3.2 seconds length. The text narrations are chosen as the 3 narration sentences that are nearest in time. In each batch, videos and their corresponding narrations are sampled as positive pairs, while videos and narrations from different sources are sampled as negative pairs.

Model details: For the video-modality, we used S3D-G [39] as the 3D CNN backbone for extracting video features. While it is possible to use larger models with better performances such as TSM [22], we choose S3D-G particularly because we would like to demonstrate that with our method, this considerably smaller model can still be trained to achieve comparable downstream task performances to large models. This is also important because smaller models like S3D-G is much more suitable for terminal device applications, which has limited computational power and on-chip memory. For the text-modality we use T5-Encoder [32] pre-trained on C4 Corpus to generate contextualised word representations. Whole vector embedding used in coarse-grained loss are generated by applying an average-pooling and linear layer to the video feature map, and a self-attention pooling [32, 34] to the contextualised word representations. The dimension of the joint video-text embedding space is determined as 512 via hyper-parameter search.

Training Setup: For self-supervised pre-training, we use a server with 8 Nvidia V100 GPUs, Intel Xeon CPU and 128 GB system memory usage limit. For fine-tuning on downstream tasks, we use a machine with 2 Nvidia Geforce 2080Ti GPUs, Intel-i9 CPU and 64 GB system memory. For self-supervised pre-training, we use a batch size of 256, and distribute the training across the 8 GPUs.

4.2 Action Recognition

Method	Dataset(Duration)	Arch.	Params.	Hardware	UCF101
MIL-NCE [29]	HTM(15y)	S3D	9.1M	64×TPU	91.3
XDC [8]	IG65M(21y)	R(2+1)D-50	46.9M	-	94.2
GDT [30]	IG65M(21y)	R(2+1)D-18	33.3M	64×GPU	95.2
MMV [10]	HTM+AS(16y)	TSM50×2	93.9M	64×TPU	95.2
Ours	HTM(15y)	S3D	9.1M	8×GPU	94.3

Table 1: Action Recognition Accuracy of downstream task fine-tuning of self-supervisedly pre-trained models. We also compare the amount of data used, model’s number of parameters, and hardware platform used for self-supervised pre-training.

We first evaluate the learnt representation on the downstream task of Action Recognition. We choose UCF101 [33] dataset, one of the most popular dataset for action recognition. We evaluate our method in the fine-tuning setting. We add a linear classifier on top of the pre-trained S3D video module, and then train on UCF101 with a smaller learning rate for the S3D module. We use learning rate of 10^{-3} for the linear classifier and 10^{-4} for the S3D parameters. We additionally apply weight decay of 10^{-5} and data augmentations including random cropping, horizontal flips and colorisation during training. For fine-tuning, we used a batch size of 64. Table 1 shows the classification accuracy of our fine-tuned model compared against previous state-of-the-art self-supervised learning methods. Our method achieves accuracy on-par with these methods, while using a smaller dataset, a much less number of parameters and considerably less pre-training hardware requirement. Note that TPU is approximately 2 times faster than Nvidia V100 GPU, and has 128GB memory, which is 4 times of V100’s 32GB memory size.

Method	Dataset(Duration)	Video Model Params	R@ 1
HowTo100M [24]	HTM(15y)	9.1M	14.9
NoiseEst [9].	IN+K400+HTM(15y+)	104M	17.4
MMT [14]	Multiple(28y+)	133.3M	26.6
SSB [13]	IN+IG65M+HTM(36y+)	101.4M	30.3
Ours	HTM(15y)	9.1M	27.1

Table 2: Text-to-Video Retrieval Performance on MSRVT dataset. We also compare the amount of data used and the parameter counts of video models.

4.3 Text-to-Video Retrieval

We further evaluate our method on the downstream task of text-to-video retrieval. We choose MSRVT as the downstream dataset due to its popularity. We evaluate our method in the fine-tuning setting, meaning that we train both the pre-trained video and text encoders on the downstream task. We use a learning rate of 10^{-3} and weight decay of 10^{-4} , and a batch size of 32. To evaluate the performance, we use Recall at K (R@K) and Median Rank(MedR). We only show R@1 result in Table 2 due to space limitation, and leave other R@K and MedR results in Appendix B. We compare against previous the SOTA Self-Supervised Learning pre-training method tested on MSRVT dataset. Our method, while using fewer data and a smaller number of parameters, achieve better R@1 than all compared methods except SSB, which uses double amount of pre-training data and 10 times larger model.

4.4 Temporal Action Detection

Method	0.3	0.4	0.5	0.6	0.7
TSN(Supervised K400)	54.5	47.6	40.2	30.8	23.4
OURS(SSL HTM)	56.7	49.8	42.1	32.0	22.8

Table 3: Temporal Action Detection result of G-TAD [40] using different pre-trained feature extraction module. Results are measured as mean Average Precision (mAP) at different IoU threshold ranging from 0.3 to 0.7. The first row shows the G-TAD’s original performance using TSN network supervisedly pre-trained on K400 dataset for feature extraction. SSL denotes self-supervised pre-training.

In this section we evaluate how well can the learnt representation be utilised for Temporal Action Detection (TAD) downstream task. In Temporal Action Detection task, a model predicts not only action classes, but also their start and end time. Most TAD methods [23, 40] uses a frozen feature extraction module to extract features for each frame of the video, and process the features using post-processing networks such as Boundary Matching Network [23] and graph neural networks [40]. In this experiment, we use our pre-trained S3D model to extract frame features, and test the quality of extracted features with G-TAD [40], one of the SOTA methods. In the original implementation, G-TAD uses TSN [57] supervisedly pre-trained on K400 dataset to extract video features for both RGB and optical flow video streams. In table 3, we compare our extracted features against TSN features by swapping the TSN RGB features with our RGB features extracted using our SSL model. We keep the same training setup as in the original work [40]. We report the mean Average Precision (mAP) at

different Intersection over Union (IoU) thresholds. Our method pre-trained on HowTo100M dataset surpasses the supervised pre-training features for all IoU thresholds except for 0.7.

4.5 Ablation Studies

In this section, we perform ablation studies to investigate the effectiveness of various design choices in our method. We use Downstream task performance on UCF101 Action Recognition dataset as the proxy metric for the quality of learnt representation. Specifically we perform ablation studies on dimensions of joint embedding space and text encoders.

Dimensions	UCF101
512	94.3
1024	93.8
6144 (norm)	91.3

Table 4: Dimensions of embedding space.

Text Encoders	UCF101
MLP	92.4
BERT	88.9
T5	94.3

Table 5: Different Text Encoders.

Dimensions of Joint Embedding Space: In Table 4 we perform an ablation study on the dimensions of the joint embedding space for sub-part pairs. While in our implementation embedding is not normalised to a unit sphere as in some other works [9, 19], we additionally tested using normalised embedding of dimension 6144, as shown in Table 4. The result is considerably worse than unnormalised counter-parts.

Text Encoders: In Table 5 we tested three different variations of the text encoders, which are MLP as in [25], BERT [64] and T5 [62]. BERT, a popular transformer-based model, performs worse than MLP. This is also observed by [25], who hypothesise that this is possibly because of domain differences in the pre-training dataset. T5, which varies slightly the BERT model but is trained on a different Corpus, has the best performance. We hypothesise that this is because the C4 corpus [62] is more aligned with the form of natural language text narrations found in the instructional video datasets.

Batch Sizes: In Table 6 we tested the performance with different pre-training batch sizes. While it is generally observed that larger batch sizes usually lead to better self-supervised training results, unfortunately we are not able to test larger batch sizes due to limited GPU memory.

Loss Functions: In Table 7 we show ablation study results of different loss functions. Here “MIL-NCE” is the typical Multi-Instance NCE loss function, as used in [25]. “FG-MMSSL” is our proposed loss function as in Equation 3, while “FG-MMSSL-No-Attn” is the loss function without the attention scaling terms, and with the dot-product term squared, as

Batch Size	UCF101
64	91.8
128	93.5
256	94.3

Table 6: Different Batch Sizes

Loss Function	UCF101
MIL-NCE	91.3
FG-MMSSL-No-Attn	92.1
FG-MMSSL-No-Inv	93.8
FG-MMSSL	94.3

Table 7: Different Loss Functions

discussed in Section 3.3. Note that squaring the dot-product exponential term is equivalent to temperature annealing as performed by other SSL methods [10, 9, 19]. It can be observed that adding an attention scaling mechanism significantly boost the UCF101 downstream task performance. Computing sub-pair loss without attention mechanism, as in “FG-MMSSL-No-Attn” experiments, there is only a 0.8% increase in accuracy, which can be mostly attributed to the more powerful T5 Text Encoder used. In “FG-MMSSL-No-Inv” we replace the $\frac{1}{1+a_{i,j}}$ negative pair attention term in Equation 3 with $a_{i,j}$. We hypothesise that having the attention mechanism, which scales the dot-product term separately, is effective because:

- The Softmax attention mechanism induces sparsity with high weighting for only one or a few pairs while low weighting for the other pairs. In many scenes, the text descriptions usually only have a high correlation with a particular part of the image/video, this sparsity inducing mechanism further reduces the contribution of the majority noisy pairs.
- The inverse scaling term $\frac{1}{1+a_{i,j}}$ for the negative pairs works by decreasing the contribution of correlated pairs in the negative pair part of the loss function, so that correlated pairs will be less pushed apart. This is further shown in this ablation experiment, as “FG-MMSSL-No-Inv” has lower downstream task performance than “FG-MMSSL”.

5 Conclusion

In this work we developed FG-MMSSL, a multi-modal self-supervised learning method that reduces noisy learning signals by adjusting pair’s weighting in the loss function using an attention-based mechanism. We show that our method can train the smaller S3D model using smaller datasets and less hardware to achieve downstream performances on par with the state-of-the-art. Training smaller models to achieve the same performance as their larger counter-parts is particularly important for device-side implementations, where computational resources are very limited. With this work, we hope to spark more interest in more efficient and environmentally-friendly self-supervised learning for low-computational-resource applications.

References

- [1] Jean-Baptiste Alayrac, Adrià Recasens, Rosalia Schneider, Relja Arandjelović, Jason Ramapuram, Jeffrey De Fauw, Lucas Smaira, Sander Dieleman, and Andrew Zisserman. Self-supervised multimodal versatile networks. *arXiv preprint arXiv:2006.16228*, 2020.
- [2] Humam Alwassel, Dhruv Mahajan, Bruno Korbar, Lorenzo Torresani, Bernard Ghanem, and Du Tran. Self-supervised learning by cross-modal audio-video clustering. *arXiv preprint arXiv:1911.12667*, 2019.
- [3] Elad Amrani, Rami Ben-Ari, Daniel Rotman, and Alex Bronstein. Noise estimation using density estimation for self-supervised multimodal learning. *arXiv preprint arXiv:2003.03186*, 2020.
- [4] YM Asano, C Rupprecht, and A Vedaldi. Self-labelling via simultaneous clustering and representation learning. In *International Conference on Learning Representations*, 2019.

- [5] Yuki M. Asano, Mandela Patrick, Christian Rupprecht, and Andrea Vedaldi. Labelling unlabelled videos from scratch with multi-modal self-supervision. In *Neural Information Processing Systems (NeurIPS)*, 2020.
- [6] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed, and Michael Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. *Advances in Neural Information Processing Systems*, 33, 2020.
- [7] Sagie Benaim, Ariel Ephrat, Oran Lang, Inbar Mosseri, William T Freeman, Michael Rubinstein, Michal Irani, and Tali Dekel. Speednet: Learning the speediness in videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9922–9931, 2020.
- [8] Honglie Chen, Weidi Xie, Triantafyllos Afouras, Arsha Nagrani, Andrea Vedaldi, and Andrew Zisserman. Localizing visual sounds the hard way. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16867–16876, 2021.
- [9] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PMLR, 2020.
- [10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=YicbFdNTTy>.
- [11] Hongchao Fang, Sicheng Wang, Meng Zhou, Jiayuan Ding, and Pengtao Xie. Cert: Contrastive self-supervised learning for language understanding. *arXiv preprint arXiv:2005.12766*, 2020.
- [12] Valentin Gabeur, Chen Sun, Karteek Alahari, and Cordelia Schmid. Multi-modal transformer for video retrieval. In *European Conference on Computer Vision (ECCV)*, volume 5. Springer, 2020.
- [13] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations*, 2018.
- [14] Simon Ging, Mohammadreza Zolfaghari, Hamed Pirsiavash, and Thomas Brox. Coot: Cooperative hierarchical transformer for video-text representation learning. *arXiv preprint arXiv:2011.00597*, 2020.
- [15] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 297–304. JMLR Workshop and Conference Proceedings, 2010.
- [16] Raia Hadsell, Sumit Chopra, and Yann LeCun. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’06)*, volume 2, pages 1735–1742. IEEE, 2006.

- [17] Tengda Han, Weidi Xie, and Andrew Zisserman. Memory-augmented dense predictive coding for video representation learning. *arXiv preprint arXiv:2008.01065*, 2020.
- [18] Tengda Han, Weidi Xie, and Andrew Zisserman. Self-supervised co-training for video representation learning. *Advances in Neural Information Processing Systems*, 33:5679–5690, 2020.
- [19] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9729–9738, 2020.
- [20] Dahun Kim, Donghyeon Cho, and In So Kweon. Self-supervised video representation learning with space-time cubic puzzles. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 8545–8552, 2019.
- [21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR (Poster)*, 2015.
- [22] Ji Lin, Chuang Gan, and Song Han. Tsm: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE International Conference on Computer Vision*, 2019.
- [23] Tianwei Lin, Xiao Liu, Xin Li, Errui Ding, and Shilei Wen. Bmn: Boundary-matching network for temporal action proposal generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3889–3898, 2019.
- [24] Antoine Miech, Dimitri Zhukov, Jean-Baptiste Alayrac, Makarand Tapaswi, Ivan Laptev, and Josef Sivic. Howto100m: Learning a text-video embedding by watching hundred million narrated video clips. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2630–2640, 2019.
- [25] Antoine Miech, Jean-Baptiste Alayrac, Lucas Smaira, Ivan Laptev, Josef Sivic, and Andrew Zisserman. End-to-end learning of visual representations from uncured instructional videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9879–9889, 2020.
- [26] Ishan Misra, C Lawrence Zitnick, and Martial Hebert. Shuffle and learn: unsupervised learning using temporal order verification. In *European Conference on Computer Vision*, pages 527–544. Springer, 2016.
- [27] Pedro Morgado, Ishan Misra, and Nuno Vasconcelos. Robust audio-visual instance discrimination. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12934–12945, 2021.
- [28] Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018.
- [29] Andrew Owens and Alexei A Efros. Audio-visual scene analysis with self-supervised multisensory features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 631–648, 2018.

- [30] Mandela Patrick, Yuki M Asano, Polina Kuznetsova, Ruth Fong, João F Henriques, Geoffrey Zweig, and Andrea Vedaldi. Multi-modal self-supervision from generalized data transformations. *arXiv preprint arXiv:2003.04298*, 2020.
- [31] Mandela Patrick, Po-Yao Huang, Yuki Asano, Florian Metze, Alexander G Hauptmann, Joao F. Henriques, and Andrea Vedaldi. Support-set bottlenecks for video-text representation learning. In *International Conference on Learning Representations*, 2021. URL <https://openreview.net/forum?id=EgoXe2zmhrh>.
- [32] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67, 2020.
- [33] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.
- [34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017.
- [35] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018. URL <https://openreview.net/forum?id=rJXMpikCZ>.
- [36] Jiangliu Wang, Jianbo Jiao, and Yun-Hui Liu. Self-supervised video representation learning by pace prediction. In *European Conference on Computer Vision*, pages 504–521. Springer, 2020.
- [37] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *European conference on computer vision*, pages 20–36. Springer, 2016.
- [38] Yongqiang Wang, Abdelrahman Mohamed, Due Le, Chunxi Liu, Alex Xiao, Jay Mahadeokar, Hongzhao Huang, Andros Tjandra, Xiaohui Zhang, Frank Zhang, et al. Transformer-based acoustic modeling for hybrid speech recognition. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6874–6878. IEEE, 2020.
- [39] Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking spatiotemporal feature learning: Speed-accuracy trade-offs in video classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 305–321, 2018.
- [40] Mengmeng Xu, Chen Zhao, David S Rojas, Ali Thabet, and Bernard Ghanem. G-tad: Sub-graph localization for temporal action detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10156–10165, 2020.
- [41] Richard Zhang, Phillip Isola, and Alexei A Efros. Colorful image colorization. In *European conference on computer vision*, pages 649–666. Springer, 2016.

- [42] Richard Zhang, Phillip Isola, and Alexei A Efros. Split-brain autoencoders: Unsupervised learning by cross-channel prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1058–1067, 2017.
- [43] Luowei Zhou, Chenliang Xu, and Jason Corso. Towards automatic learning of procedures from web instructional videos. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018.

A MMSSL Analysis for other modality

While in Section 3.2 we analysed the case for video modality, here we perform the same analysis on text modality. For Text modality, the input \mathbf{y} is usually a sequence of tokens that index different words in a specific natural language. The input is often processed in an encoding pipeline, which are implemented with architectures including Recurrent Neural Networks, Convolutional Neural Networks and Transformers. The output of this processing pipeline is a sequence of contextualised embedding feature vectors $\mathbf{l} = (l_1, \dots, l_N)$. In most Multi-Modal Self-Supervised Learning works [10, 9, 24, 25], the sequence is first pooled using a Max-Pooling operation and generate a single feature vector representing the whole sequence. The feature vector is then further projected into the multi-modality embedding space using a linear layer $W \times \text{MaxPool}(\mathbf{l}) + b$. In this case for text modality, we can then rewrite the dot-product terms in equation 1 as:

$$e^{f_1(x)^T f_2(y)} = e^{f_1(x)^T (W \times \text{MaxPool}(\mathbf{l}) + b)} = e^{f_1(x)^T W \times \text{MaxPool}(\mathbf{l}) + f_1(x)^T b} \quad (6)$$

It is not possible to break down *MaxPool* into individual components for each feature vector l_i in a similar fashion to *AvgPool* in Equation 6. However we can still perform some analysis. *MaxPool* takes a sequence of same-dimension vectors, and return the largest value in each dimension. This is equivalent to finding a hyper-cube that tightly bounds all vectors in the sequence, and return the vector of the vertex in the domain where all dimension axes are positive. We first make the assumption that in the input text \mathbf{y} there are words that are not correlated with the video modality (which is a very reasonable assumption in narrated videos). Unless *MaxPool* is guaranteed to return values only from feature vectors corresponding to the video-correlated words, the gradient will propagate back to the feature vectors of video-uncorrelated words. While *MaxPool* cannot make this guarantee, video-uncorrelated words will introduce noise in training in the same way as discussed in Section 3.2. This is particularly the case at the start of the training, where word feature vectors are generated from random embedding.

B MSRVTT Full Result

Table 8 shows the full Text-to-Video retrieval results on MSRVTT datasets. The metrics include Retrieval rate at (1, 5, 10) and Median Rank.

Method	R@1	R@5	R@10	MR
HowTo100M [24]	14.9	40.2	52.8	9
NoiseEst [9].	17.4	41.6	53.6	8
MMT [10]	26.6	57.1	69.6	4
SSB [51]	30.3	58.5	69.3	3
Ours	27.1	57.4	69.1	4

Table 8: Text-to-Video Retrieval Performance on MSRVTT dataset (Full Results) $R@K$ denotes retrieval at K while MR denotes Median Rank.