Multimodal Phased Transformer for Sentiment Analysis

Junyan Cheng[*]
University of Southern California,
Los Angeles, CA, USA
junyanch@usc.edu

Iordanis Fostiropoulos[*]
University of Southern California,
Los Angeles, CA, USA
fostirop@usc.edu

Barry Boehm
University of Southern California,
Los Angeles, CA, USA
boehm@usc.edu

Mohammad Soleymani
USC Institute for Creative Technologies,
Los Angeles, CA, USA
msoleyma@usc.edu

Abstract

Multimodal Transformers achieve superior performance in multimodal learning tasks. However, the quadratic complexity of the self-attention mechanism in Transformers limits their deployment in low-resource devices and makes their inference and training computationally expensive. We propose multimodal **Sparse Phased Transformer** (SPT) to alleviate the problem of self-attention complexity and memory footprint. *SPT* uses a sampling function to generate a sparse attention matrix and compress a long sequence to a shorter sequence of hidden states. *SPT* concurrently captures interactions between the hidden states of different modalities at every layer. To further improve the efficiency of our method, we use *Layer-wise* parameter sharing and **Factorized Co-Attention** that share parameters between *Cross Attention Blocks*, with minimal impact on task performance. We evaluate our model with three sentiment analysis datasets and achieve comparable or superior performance compared with the existing methods, with a 90% reduction in the number of parameters. We conclude that (SPT) along with parameter sharing can capture multimodal interactions with reduced model size and improved sample efficiency.

1 Introduction

The objective of multimodal sentiment analysis is to identify the polarity of one's attitude toward an entity through multimodal inputs such as audio, video, and text. For many applications such as personal assistants, social robots and virtual agents, the efficiency and scalability of a method are as important as accuracy. Such applications can have limited computational resources or large-scale deployment requirements. Multimodal understanding of constructs, such as sentiment, requires capturing information available in every modality in addition to their potential interactions, *e.g.*, an exaggerated smile combined with negative sentiment in language might signal irony. Modeling these interactions efficiently is still an open challenge (Baltrušaitis et al., 2018). Some work on this topic use the different networks for each modality followed by fusion methods, like concatenation (Tsai et al., 2019; Hazarika et al., 2020; Pan et al., 2020) and outer-product (Zadeh et al., 2017), to model the interaction of multimodal representations which largely increases the dimensionality of representations thus increasing computational cost. Rahman et al. (2020) rely on large pre-trained models, like BERT (Devlin et al., 2018) and XLNet (Yang et al., 2019). The computational cost of such approaches is high due to the over-parameterization of the models. And Transformer-based methods (Tsai et al., 2019; Rahman et al., 2020) suffer from quadratic complexity of self-attention.

The existing multimodal sentiment analysis datasets are rather small due to the laborious labeling process. The development of the existing datasets, such as (Zadeh et al., 2016; Bagher Zadeh et al., 2018), involves data curation and annotation by multiple annotators. The limited dataset size raises the risk of over-fitting for over-parameterized models which motivated building models that can be trained with fewer data.

Recent work, such as (Child et al., 2019; Beltagy et al., 2020; Zaheer et al., 2020), improve the efficiency of Transformers through *Sparse-Attention*. Compared to the full attention that calculates attention for all pairs of input elements, sparse attention only computes attention for a subset of element pairs. As a consequence, each element from one sequence attends only to a limited number of elements in the source sequence. Other work *reduce attention matrix size* by iterative processing only a shorter segment of the original long sequence at a time (Dai et al., 2019; Rae et al., 2020).

In this paper, we propose a ***Sparse Phased atten-***

[*] equal contribution

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 2447–2458
November 7–11, 2021. ©2021 Association for Computational Linguistics

tion (SP) mechanism that uses a *sampling function* to compress a longer input sequence to a shorted sequence of hidden states and improve the efficiency of the attention computation. Multimodal *SPT* can capture multimodal interactions through a "Concurrent" network structure rather than a "Serial" structure of previous work (Tsai et al., 2019; Rahman et al., 2020; Huang et al., 2020; Pan et al., 2020). We improve the efficiency of SPT through parameter sharing and a **Factorized Co-Attention**. We perform extensive experiments evaluating the performance of SPT on multimodal sentiment analysis and an ablation study on its structure, sampling function, parameter sharing approaches, and SP-Block. We compare our method with the state-of-the-art efficient Transformers, *i.e.*, Performer (Choromanski et al., 2020). Our experimental results show that our model is able to achieve minimal or no performance loss with a significant reduction in model size. Other efficient Transformer-based approaches with linear efficiency result in a larger degradation of the performance. In comparison with the previous work on multimodal sentiment analysis, we reduce memory use in addition to training and inference time, with complexity decreasing from quadratic to linear, with only 10% of the parameters.

The main contributions of this work are as follows.

- We introduce and evaluate a **SP-Block** that uses a *sampling function* and a short sequence of hidden states to attend to and compress a longer sequence. SP-Attention creates a sparse attention matrix that improves both computational and sampling efficiency.

- We propose **Multimodal SP-Transformer** that uses a *concurrent* structure of blocks in each sub-layer to allow multimodal signal to interact within every layer. *SPT* uses *Input Attention* on the source input sequence, *Cross-Attention* on the hidden state pairs of different modalities and *Self-Attention* on the hidden states of each modality.

- We leverage **Factorized Co-Attention** that use a factorized form of the attention computation based on an affinity matrix to further reduce the number of parameters for the cross attention block (*Co-SP*). And we further the improve efficiency of *SPT* by parameter sharing across all layers.

The code and data are publicly available in `https://github.com/chengjunyan1/SP-Transformer`.

2 Related work

Sentiment analysis Multimodal sentiment analysis involves leveraging the information from multiple modalities, *e.g.*, text and vision, to recognize the polarity of expressed sentiment. Most of the existing work focuses on recognizing sentiment expressed in video recordings from social media reviewing products or movies (Zadeh et al., 2016; Kossaifi et al., 2019; Wöllmer et al., 2013).

Recent work on multimodal sentiment analysis has focused on the application of Transformer architectures. Tsai et al. (2019) introduces pairwise cross-modal attention on Transformers for multimodal sentiment analysis on audio, video, and text. Our model's architecture follows a similar design that first encodes unimodal inputs, then models cross-modal interactions and finally fuses multimodal information. The main difference is that our model enables a *concurrent* way to implements those steps. Hazarika et al. (2020) project multimodal input into *modality-invariant* and *modality-specific* spaces, and use a Transformer encoder on the concatenated projected representations. Rahman et al. (2020) use pre-trained Transformers like BERT (Devlin et al., 2018), XLNET (Yang et al., 2019) on a large corpus and perform transfer-learning for multimodal sentiment analysis.

Multimodal learning Previous work use recurrent neural networks (RNN) or convolutional neural networks (CNN) on each modality and perform model-based, *e.g.*, kernel-based fusion, with graphical models, and neural networks, and model-agnostic fusion, *e.g.*, early, late or hybrid (Baltrušaitis et al., 2018). Wang et al. (2019) fuse multimodal representations with a *Gated Modality-mixing Network*, that model the fine-grained structure of non-verbal subword sequences and a *Multimodal Shifting* mechanism, that dynamically shifts word representations based on non-verbal cues. Pham et al. (2019) trains a sequence-to-sequence RNN to jointly perform inter-modality translations and sentiment analysis, the encoder output is a joint multimodal representation that is used for sentiment detection. Tensor fusion networks (TFN) use the outer product of representations for each modality concatenated with a constant value of "1" to generate a joint representation (Zadeh et al., 2017). Liu et al. (2018) propose to decompose the weight from the fusion layers into low-rank factors to reduce the large number of parameters in *TFN*. Zadeh et al. (2018) use a system of LSTMs to learn modal-specific interactions, learn the cross-modal interactions with an attention mechanism and finally apply a *Multi-view Gated Memory* that fuses the multimodal representations through time. Alayrac et al. (2020) project multimodal features to *fine-granularity* and *coarse-granularity* "spaces" through Multi-Layer-Perceptron (MLP) Networks. Audio and video are aligned with text features in *coarse-grained space*, while audio and video features are aligned in *fine-grained space*. Unlike the previous methods that directly apply transformations on the multimodal inputs, we use a small sequence of hidden states to capture the features from multimodal inputs thus preserving raw input information while improving efficiency.

In the most similar work to ours, Jaegle et al. (2021) distill information from an input sequence to a fixed length of hidden states with an Autoregressive Transformer.

Efficient Transformers One of the drawbacks of the Transformer architecture is the computational cost and the memory footprint of the self-attention mechanism. A number of efforts have been made to make Transformers more efficient (Child et al., 2019; Beltagy et al., 2020; Kitaev et al., 2020; Zhou et al., 2021; Zaheer et al., 2020). Such work use sparse attention to selectively attend to pairs of elements and lead to a reduction in memory use and computational complexity of the attention mechanism. A notable example, Performer, (Choromanski et al., 2020), improve the efficiency of the attention computation through "unbiased" and low-rank approximation of the attention matrix.

There are other attempts for reducing the sequence length to improve computational efficiency. Dai et al. (2019) introduce "segment-level recurrence" that recurrently use the previous segment and current segment. Rae et al. (2020) extend this idea and compresses multiple previous segments into memory vectors. The hidden states in our method are based on a similar idea that caching the information from the input sequence in a shorter sequence can improve efficiency. We also applied the idea of sparse attention to achieve further improvements.

3 Method

The proposed method is an extension of the Transformer architecture for improved efficiency in multimodal learning. In this section, we will introduce the basic building block of our model, *i.e.*, Sparse-Phased-Block (SP), and show how we extend *SP* in the context of multimodal learning to define *SP-Transformer* with *Input Attention*, *Cross Attention* and *Self Attention* sub-layers. We also leverage *Factorized Co-Attention* and parameter sharing for further optimizations of *SP-Transformer*.

Each *SP-Block* uses a sampling function to generate sparse attention patterns that guide a sequence to selectively compress information from elements of another sequence as shown in Figure 2. We use SP-Block to build our *Multimodal SP-Transformer*, in each layer, we first use an SP-Block that executes *Input attention* for each modality and compress information from each unimodal input sequence to the hidden states. The hidden state sequences for each pair of two modalities interact through *Cross Attention* by a *Co-SP-Block* using *Factorized Co-Attention*. Finally, the cross-modal information for each modality is fused by summation and distilled with *Self Attention* using an SP-Block. The full model is presented in Figure 3.

3.1 Sparse Phased Block

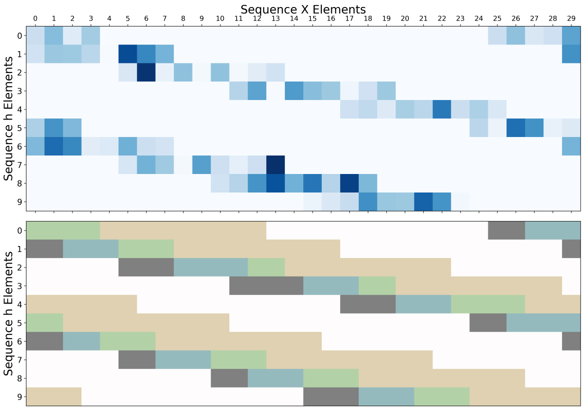

Figure 1: A sequence of hidden state h with length 10 sample from a sequence X of length 30. From top to down: (1) Sparse attention pattern created by *Mixed* sampling function. (2) Attention mask for the same function and for four layers of SPT. Different color is used for each layer and top layer overlap bottom layers.

Sparse Phased Block (**SP-Block**) accept a sequence X and a sequence of hidden state h as input and use **sampling function** ψ to compress X to h by selective attention between the two. ψ create

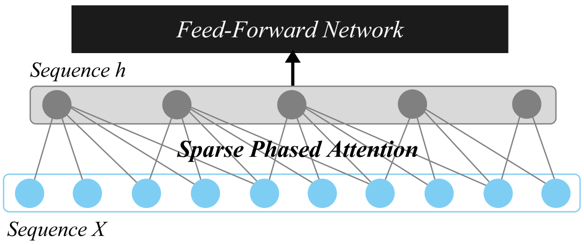

Figure 2: *SP-Block* samples from a sequence X with hidden state h and with a *fixed* sampling function with sampling length $r = 2$. The hidden states of h attend at most to five input states of X. In a *mixed* sampling, the sampling interval for each hidden state would shift with a distance that depends on the layer (*sliding*), a *periodic* function with its index as parameter, and *random* perturbation, which makes it sample in a "dynamic" way as opposed to a static (fixed) sampling.

a sparse attention mask for which h is used as the *Query* vector and X is *Key* and *Value*.

$$SP\text{-}Block(h, X) = FFN(W^O \|_{i=0}^{H} head_i + h)$$
$$head_i = (G \odot \sigma(\frac{hW_i^Q (XW_i^K)^T}{\sqrt{d_k}}))XW_i^V$$

where σ is softmax function, $\|$ is concatenation, $W_i^Q \in \mathbb{R}^{d_{model} \times d_k}, W_i^K \in \mathbb{R}^{d_X \times d_k}, W_i^V \in \mathbb{R}^{d_X \times d_v}, W^O \in \mathbb{R}^{d_{model} \times Hd_v}$ are parameter matrices. H is number of heads, d_{model} is the dimension of the model and the query sequence h, d_X is the dimension of the key and the value sequence X. $FFN(x) = W_2 ReLU(W_1 x + b_1) + b_2 + x$ is Feed-Forward Network (FFN) (Vaswani et al., 2017) where $W_1 \in \mathbb{R}^{d_{model} \times d_{ff}}, W_2 \in \mathbb{R}^{d_{ff} \times d_{model}}$ are parameter matrices, $b_1 \in \mathbb{R}^{d_{ff}}, b_2 \in \mathbb{R}^{d_{model}}$ are bias terms and include a residual connection to x.

The SP-Block is illustrated in Figure 2. For each block we apply layer normalization prior to FFN and Attention function. Sampling function ψ can define single or multiple interval in X and is used to compute a boolean attention mask G. Every element $h_i \in h$ attend to element $X_j \in X$ only if $j \in \psi(i)$ and $G_{ij} = \mathbb{1}_{\psi(i)}(j)$. We experiment with four sampling functions that use **Sliding**, **Periodic**, **Fixed** and **Random** attention patterns.

Sampling function map every hidden state h_i to an interval in X with a *sampling length* $r \in \mathbb{N}$ such that $\psi(i) = (\rho(i) + \phi)\%L_X$ where $\rho(i) = [\frac{L_X}{L_h}i - r, \frac{L_X}{L_h}i + r]$ and $\phi_f = 0$ for **Fixed** sampling function. A convolution operation is similar to the *Fixed* sampling found in Figure 2. **Sliding** sampling function shift the same interval at every layer such that that $\phi_s(i) = \alpha\lambda$ with magnitude α. **Periodic** sampling function map h_i to multiple intervals that periodically span X such that $\phi_p(i) = L_X sin(\beta \times i)$ with magnitude β. **Random** sampling function define a random interval in X such that $\phi_r = U(-\gamma, +\gamma)$ and window γ. **Mixed** sampling combine the above pattern such that $\phi_m = \phi_s + \phi_p + \phi_r$ and improve performance compared to each sampling function individually. An example of attention mask generated by sampling function is visualized in Figure 1.

Implementation of efficient sparse computation in GPU is known to be challenging (Zaheer et al., 2020). The sparse pattern introduced can be optimized for GPU, similar to previous work (Beltagy et al., 2020; Child et al., 2019; Zaheer et al., 2020) with an additional speed-up for custom CUDA kernels (Beltagy et al., 2020; Child et al., 2019). In our experiments, we do not consider custom CUDA kernels a discussed in Section 5.

3.2 Multimodal Sparse Phased Transformer

Multimodal **SP-Transformer**, is a stack of *SP-Transformer layers* composed of *Input Attention*, *Cross Attention* and *Self Attention* sub-layer applied in the same order. All blocks are identical in computation and are defined by *SP-Block*. *SP-Transformer* layer accept multiple sequence X_m, where $m \in M$ is the set of all modalities as well as **hidden states** h_m^λ for each modality at layer λ. The layer output updated hidden states $h_m^{\lambda+1}$. The first layer use learnable embedding h_m^0. At every layer *Input Attention* attend to the original signal X_m with the hidden states from the previous layer h_m^λ to compute updated hidden states for modality m, $\hat{h}_m^{\lambda+1}$. *Cross Attention* is applied on the output of the two *Input Attention* blocks of different modalities. For every modality, *Cross Attention* attend to the hidden states between $m \rightarrow m'$, $\forall m' \in M \backslash \{m\}$. We extend *Cross Attention* to *Co-SP-Block* that shares the parameters for attention between the hidden states of modality $m' \rightarrow m$ and $m \rightarrow m'$. We describe *Co-SP-Block*, in detail, later in this section. Finally, we sum the *Cross Attention* hidden states for modality $m \rightarrow \forall m'$ and apply a *Self Attention* mechanism in the final vector that represent the hidden states learned for modality m defined as $h_m^{\lambda+1}$. The output of this layer can be fed to another layer or be used in a downstream task. The architecture is illustrated in Figure 3.

Input Attention which compresses each unimodal input sequence into hidden states is defined as follows.

$$\hat{h}_m^{\lambda+1} = SP\text{-}Block(h_m^\lambda, X_m)$$

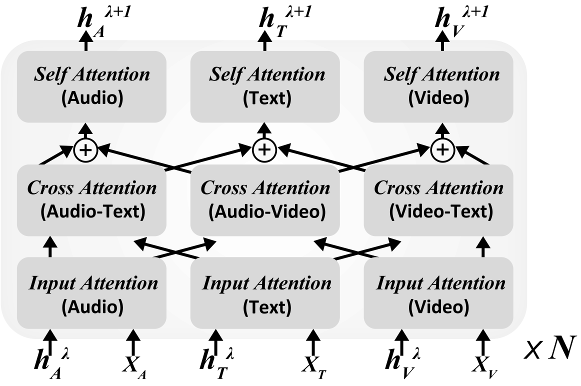

Figure 3: A trimodal SP-Transformer for audio A, video V, and text T, with N layers to update hidden states h_m. *SP-Block* is indicated with grey rectangles.

Cross Attention models cross-modal interaction between hidden state sequences for two modalities, as follows.

$$\bar{h}_m^{\lambda+1} = \sum_{m' \in M} SP\text{-}Block(\hat{h}_m^{\lambda+1}, \hat{h}_{m'}^{\lambda+1})$$

Self Attention refines the representation fusing cross-modal information for each modality.

$$h_m^{\lambda+1} = SP\text{-}Block(\bar{h}_m^{\lambda+1}, \bar{h}_m^{\lambda+1})$$

The use of hidden states is the main difference between our method and the previous Transformer-based methods (Tsai et al., 2019; Rahman et al., 2020; Hazarika et al., 2020). Information from longer input sequences is absorbed by a shorter hidden state sequence *iteratively* for every layer instead of only the first layer (Rahman et al., 2020; Tsai et al., 2019; Huang et al., 2020; Pan et al., 2020) or *recurrently* on segments of the input sequence (Dai et al., 2019; Rae et al., 2020). In our experiments, we constraint the length of a hidden state sequence $L_{h_m} = \frac{L_{X_m}}{S}$ where S is a hyperparameter to control the compression ratio.

Previous work (Tsai et al., 2019; Rahman et al., 2020; Huang et al., 2020; Pan et al., 2020) apply a set of sub-layers serially multiple times with the output of one stack as input to the next. We perform experiments on a *Serial Structure* and use Input Attention sub-layers $\times N \rightarrow$, Cross attention sub-layers $\times N \rightarrow$ and Self Attention sub-layers $\times N$, available in Appendix D. For each modality, a **Concurrent Structure** use Input Attention blocks $\rightarrow$, Cross Attention blocks $\rightarrow$ and Self Attention blocks with interaction happening only in the Cross Attention sub-layers. The same process is repeated N times. SPT uses *concurrent structure* that fuses cross-modal information with summation. We also experiment with a variant of a *concurrent structure* that uses concatenation (see the Appendix C).

3.3 Optimized Sparse Phased Blocks

Cross attention sub-layer models the bimodal interaction. We would need two SP-Blocks to model the interaction between modalities $A \rightarrow B$ and $B \rightarrow A$. The number of *SP-Block*s required to model pair-wise interactions has a quadratic growth with respect to the number of modalities. *Factorized Co-Attention* reduces the number of parameters by half and shares a *SP-Block* for a given pair of modalities without accuracy loss. We extend the idea of Co-Attention (Lu et al., 2016) to **Factorized Co-Attention** where an affinity matrix C represents the distance between two sequence X and Y with $C = d(X,Y) = XWY^T = d(Y,X)^T = (YW^TX^T)^T$, for which $W = W^QW^{K^T}$. **Co-SP-Block** applies *Factorized Co-Attention* and shares trainable parameters (FFN, W^Q, W^K, W^V and W^O) between two *SP-Blocks*. We omit the multi-head notation for clarity.

$$\begin{aligned} h_{mm'} &= Co\text{-}SP_m(\hat{h}_m, \hat{h}_{m'}) = \\ &FFN(W^O G \odot \sigma(C)\hat{h}_m W^V) \\ h_{m'm} &= Co\text{-}SP_{m'}(\hat{h}_{m'}, \hat{h}_m) = \\ &FFN(W^O G^T \odot \sigma(C^T)\hat{h}_{m'} W^V) \end{aligned}$$

The pairs of modalities in cross attention have a quadratic growth (i.e., the pair-wise hidden state sequences for a *Cross Attention* sub-layer will be $|\mathcal{M}|(|\mathcal{M}|-1)$). Previous work (Tsai et al., 2019; Hazarika et al., 2020) concatenate the representations of modalities to fuse cross-modal information. In contrast, we add the pair-wise hidden states for each modality, which reduces the size of the model and reduces its complexity.

SP-Transformer also shares parameters across layers which have shown to be effective by Lan et al. (2020) and Jaegle et al. (2021). We did an ablation experiment for all parameter sharing patterns, the results are presented in Section 4.3.

4 Experiment

We introduce the experimental setup, baseline methods and datasets. We present the results and additional evaluations through an ablation study.

4.1 Experimental setup

We evaluate our model on three multimodal sentiment and humor analysis datasets, namely, CMU-MOSI (Zadeh et al., 2016), UR-FUNNY Hasan et al. (2019) and CMU-MOSEI (Bagher Zadeh et al., 2018). CMU-MOSI (Zadeh et al., 2016) includes 2,199 short monologue video clips, CMU-MOSEI (Bagher Zadeh et al., 2018) includes 23,454 movie review video clips taken from YouTube, and UR-FUNNY (Hasan et al., 2019) comprises 8253 punchlines from TED talks. We classify each sample with a positive/negative sentiment for CMU-MOSI and CMU-MOSEI and humor/non-humor for UR-FUNNY. Additional details about the datasets can be found in Appendix A. For CMU-MOSI and CMU-MOSEI, we use the preprocessed datasets provided with the code of the Multimodal Transformers (MulT) (Tsai et al., 2019). We experiment on both aligned and unaligned data and denote with a suffix (*A*) and (*UA*) respectively. Audio and video features that have no time-aligned text features are excluded from the aligned dataset while they are preserved in the unaligned dataset. For the UR-FUNNY dataset, we use the publicly available extracted features for text (Glove), audio (COVAREP), and video (OpenFace (Baltrusaitis et al., 2018)) from Hasan et al. (2019). We report the accuracy, F1 score, and the number of trainable parameters for a model as metrics for all our experiments.

We use Tsai et al. (2019) as the baseline for CMU-MOSI and CMU-MOSEI that is state-of-the-art for publicly available features extracted with Glove (Pennington et al., 2014) for text, Facet (Littlewort et al., 2011) for video and COVAREP (Degottex et al., 2014) for audio. We compare and follow the methodology from Hazarika et al. (2020) for UR-FUNNY which is state-of-the-art for the Glove feature on text. The BERT feature reported by the same work is not publicly available and requires manual extraction and check. There are work that achieve higher performance for the aforementioned datasets but do not publish the preprocessed data or code (Sun et al., 2020; Hasan et al., 2021). It is not possible to directly compare with those methods.

We perform a grid search for some of the hyper-parameters, consistent with previous work (Tsai et al., 2019; Rahman et al., 2020; Hazarika et al., 2020; Sun et al., 2020), and empirically select the remaining ones. Our hyper-parameter settings and optimization strategy are available in Appendix B.

4.2 Results

Our method achieves comparable results in MOSI and superior results in MOSEI and UR-FUNNY datasets. **Sample efficiency** define the efficiency of a model in leveraging information from a single training sample. We follow a similar methodology as previous work (Khandelwal et al., 2019) and use multiple identical training subsets from the unaligned CMU-MOSEI dataset to compare the sample efficiency between our model and MulT. Even though, the improvement is marginal our model uses a significantly lower number of parameters and has a higher sample efficiency.

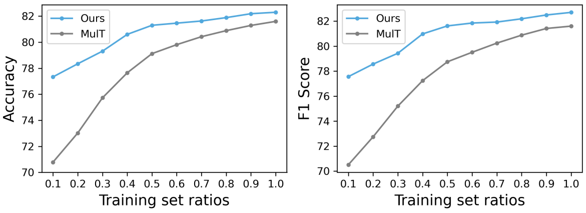

Figure 4: Sample efficiency test on unaligned CMU-MOSEI dataset in comparison with MulT. We gradually increase the size of the training set and use the same training set for both models for consistency.

We compare SPT ("Ours") with layer-wise parameter sharing, *mixed* sampling and summation for cross-modal interactions with other state-of-the-art. Our model use 154K trainable parameter which is a reduction of 90% compared to Tsai et al. (2019) and 97% for Hazarika et al. (2020). Detailed result are listed in Table 1. The reduction in parameters can also explain the improved sample efficiency of our model as shown in Figure 4.

4.3 Additional Experiments

We perform an ablation study on *SP-Transformer* and quantitatively evaluate the memory use, inference time, and training time for our model. Results of ablation study are available in Table 2.

Ablation experiment on network structure We modify and experiment with the structure of *SP-Transformer* described in Section 3.2. We experiment on multimodal interactions with a *Serial* model and two variations of the *Concurrent* model, with *summation* ("Ours") and *concatenation* on the output of *Cross Attention* sub-layer. Summation use half the parameters compared with the concatenation with nearly identical accuracy. *Concurrent structure* improves accuracy compared with a serial structure which could be due to the richer multimodal interactions at every layer.

Parameter sharing We analyze the influence of parameter sharing strategies on the model performance.

Table 1: Result on aligned and unaligned CMU-MOSI, CMU-MOSEI datasets for (1) LF-LSTM (Tsai et al., 2019) (2) Wang et al. (2019) (3) Pham et al. (2019) and UR-FUNNY dataset for (5) Zadeh et al. (2017) (6) Liu et al. (2018) (7) C-MFN (Zadeh et al., 2018) (8) Hazarika et al. (2020). RAVEN and MCTN are trained with Connectionist Temporal Classification (CTC) loss (Graves et al., 2006) on unaligned datasets.

	MOSI-A		MOSI-UA		MOSEI-A		MOSEI-UA			UR-FUNNY		
Model	**Acc**	**F1**	**Acc**	**F1**	**Acc**	**F1**	**Acc**	**F1**	**Size**	**Model**	**Acc**	**Size**
LSTM[1]	76.8	76.7	77.6	77.8	80.6	80.6	77.5	78.2	-	TFN[5]	64.7	-
RAVEN[2]	78.0	76.6	72.7	73.1	79.1	79.5	75.4	75.7	-	LMF[6]	65.2	-
MCTN[3]	79.3	79.1	75.9	76.4	79.8	80.6	79.3	79.3	-	MFN[7]	65.2	-
MulT[1]	**83.0**	82.8	81.1	81.0	82.5	82.3	81.6	81.6	1.56M	MISA[8]	68.6	5.34M
Ours	82.8	**82.9**	**81.2**	**81.3**	**82.6**	**82.8**	**82.4**	**82.7**	**154K**	Ours	**70.0**	**158K**

We perform experiments on a model that does not share parameters across layers ("Layer NS") and within cross attention sub-layer ("Cross NS"). Our results indicate that parameter sharing can decrease the model size by 71% with negligible impact on model accuracy. Layer-wise parameter sharing improves performance, this could be due to the fact that sharing reduces the risk of over-fitting. This is in accordance with the results reported in Jaegle et al. (2021).

We test two additional sharing strategies, sharing parameters between identical block types for the same modality ("Modal S") and for all *SP-Block* ("All Share"), across all layers and sub-layers. Due to the difference in the dimensionality of the sequence between each modality, we use a linear projection to map audio, video, text inputs to d_{model}. Results show that further sharing reduces the size of the model by 70% compared with our model, with a 1.5% relative reduction in model accuracy. The trade-off between accuracy and model efficiency can be adjusted depending on the use case. The results demonstrate that parameter sharing has a small effect on model accuracy, in our approach.

Sampling Function We perform experiments on the five attention mask patterns introduced in Section 3.1. "Ours" model use a *Mixed* sampling function and we experiment with a "Fixed", "Slide", "Period" and "Random" sampling function applied independently, as well as a "Fullattn." as introduced by Vaswani et al. (2017). A full attention mask is significantly slower but outperforms other sampling functions when used in isolation. A *mixed* sampling with "Ours" is a combination of Slide, Period, and Random mask which outperform full attention. At every layer, multiple hidden states will attend to the original input sequence for different intervals (sliding sampling) or overlapping intervals (period sampling) and with a regularization effect (random sampling). The structured sparsity of *mixed* sampling can introduce an inductive bias that allows hidden states to learn compressed representations from the longer input signal.

Unimodal experiment We train SPT on the CMU-MOSEI unaligned dataset on a single modality of text. Results for the unimodal setting use a suffix "U" and can be found in Table 2.

We compare SPT with the result reported by MulT ("U") for a Unimodal setting. We also train a model that replaces the Transformer block with *SP-Block* in MulT ("MulT-SP"). SPT ("U") uses an Input Attention block followed by a Self-Attention block in each layer. Results show a substantial difference in the performance for SP-Block. The difference between MulT-SP and Unimodal SPT is in the downsampling by Conv1D as opposed to the compression by Input Attention block. The advantages of SP-Block lead to a 3.3% increase in performance and a 89% reduction in parameters.

Comparison with Performer To compare our method with the state-of-the-art efficient Transformer-based architectures, we compare our method with Performer (Choromanski et al., 2020) using the same architecture from MulT with Performer layers in both multimodal ("MulP") and unimodal ("MulP (U)") setting. Results indicate that our method could improve efficiency without the loss of accuracy, unlike Performer.

Efficiency test is performed on the inference time and memory use of our model on different input sequence length and we compare to MulT, a MultiModal Performer ("MulP"). Other state-of-the-art methods, MAG (Rahman et al., 2020), MISA (Hazarika et al., 2020) use Transformer and

Table 2: Alblation study on SP-Transformer with Aligned (A) and Un-Aligned (UA) CMU-MOSEI dataset that use different attention patterns ("Fullattn.", "Fixed", "Slide", "Period", "Random"), model structure ("Serial", "Concat."), and parameter sharing strategies ("Cross NS", "Layer NS", "Modal S", "All Share"). Unimodal ("U") are model trained with only text features. [1] Multimodal Performer ("MulP") (Choromanski et al., 2020) [2] Multimodal Transformer ("MulT") (Tsai et al., 2019).

	MOSEI-A		MOSEI-UA		
	Acc	**F1**	**Acc**	**F1**	**Size**
Ours	82.6	82.8	82.4	82.7	154K
MulP[1]	82.0	81.8	81.3	81.2	1.56M
Fullattn.	82.5	82.4	82.2	82.4	154K
Fixed	82.3	82.3	82.0	82.3	154K
Slide	82.4	82.6	82.4	82.5	154K
Period	82.4	82.5	82.2	82.6	154K
Random	82.4	82.4	82.1	82.2	154K
Serial	82.3	82.3	81.9	82.2	154K
Concat.	82.6	82.9	82.4	82.6	322K
Cross NS	82.6	82.8	82.4	82.6	168K
Layer NS	82.6	82.7	82.3	82.5	545K
Modal S	81.4	81.3	81.5	81.9	70.4K
All Share	81.4	81.8	81.0	81.5	44.8K
MulT[2] (U)	-	-	77.4	78.2	430K
MulP (U)	-	-	77.2	78.1	430K
SPT (U)	-	-	80.7	80.9	45.5K
MulT-SP (U)	-	-	77.8	78.4	430K

share the same quadratic complexity with MulT. We keep the same $d_{model} = 32$ and layers $\lambda = 4$ for all models. Detailed results on inference time and memory use are found in Figure 5.

Results show that our model achieves linear complexity $O(\frac{rL}{S})$ on both memory use and inference time with respect to the sequence length L and with a slope determined by the compression ratio S and segment length r. The improvement is a result of the downsampling from the Input Attention, sparse attention from the sampling function, and a simplified model structure.

We test training time in unaligned CMU-MOSI, CMU-MOSEI, and UR-FUNNY datasets. All experiments use the largest batch size that can be executed on a single NVIDIA Tesla V100 with 16GB vRAM. Result are listed in Table 3. With a compression factor, $S = 8$ our method reduces the training time per epoch by 83%, 81%, 72% in MOSI, MOSEI, UR-FUNNY respectively compared to MulT, and 58%, 43%, 32%. Improvement in training time is a result of the auto-encoding compressive properties of SP-Block and ratio S.

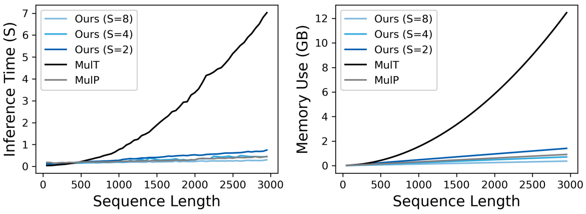

Figure 5: Multimodal Transformer ("MulT") from Tsai et al. (2019). Performer ("MulP"). SPT ("Ours") with variable compression factor S. From left to right: (1) Test on CPU inference time. (2) Test on memory use.

Table 3: Comparison between our model, MulT and Performer ("MulP") in training time (seconds per epoch) for the maximum batch size and for different compression ratios S with r fixed to 8.

	MOSI-UA	MOSEI-UA	UR-FUNNY
$S = 2$	9.8s	137.6s	119.9s
$S = 4$	4.9s	70.2s	72.6
$S = 8$	**2.5s**	**37.5s**	**48.2s**
MulT	14.5s	192.7s	171.1s
MulP	6.0s	65.4s	70.7s

5 Future work

Sampling function Other work explore complex sparse patterns like dilated sliding window (Beltagy et al., 2020) that allows the segment to be "dilated" with gaps between sampled elements, routine-based (Kitaev et al., 2020) that samples the nearest neighbors for the hidden states in the sequence, probabilistic (Zhou et al., 2021) that samples based on KL divergence, global blocks (Zaheer et al., 2020) which allows few elements sampling the entire sequence or uses trainable parameters (Tay et al., 2020; Neil et al., 2016; Hu and Qi, 2017; Mei and Eisner, 2017) which enable the model learn to select the elements to sample. We consider such sampling patterns for future work.

Hidden states are randomly initialized. Inductive bias can be introduced to further improve performance and sample efficiency. Melis et al. (2020) warmed up the hidden state in RNNs and allow interactions of the initial hidden state with the input prior to being used by the model.

Implementation methods rely on custom CUDA kernels (Beltagy et al., 2020; Child et al., 2019) to further optimize sparse computation in GPU. Our work does not implement any custom CUDA kernels, thus only achieve memory advantages from the sparse pattern. We expect that specialized GPU optimization should further improve our efficiency. Moreover, further optimization could be achieved by incorporating the factorization method from Choromanski et al. (2020).

6 Conclusions

In this paper, we propose a multimodal SP-Transformer that uses a sequence of hidden states to sample from longer multimodal input sequences. Compared with the previous Transformer-based models, our model has a reduced computational complexity through sparse attention. The concurrent structure also enables more effective capturing of the multimodal interaction, resulting in higher performance. The optimization through parameter sharing patterns leads to a significantly lighter model, with a lower number of parameters and improved sampled efficiency. As a result, the proposed model's performance is superior or comparable to the existing methods at a lower computational and memory cost. Our experiments show that our method has a good balance between accuracy and efficiency and has the potential to be deployed in real-world multimodal applications.

Acknowledgements

Research was sponsored in-part by the Army Research Office and was accomplished under Cooperative Agreement Number W911NF-20-2-0053. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Office or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein. This material is also based upon work supported in part by the U.S. Department of Defense through the Systems Engineering Research Center (SERC) under Contract No. HQ0034-13-D0004 Research Task WRT 1016 – "Reducing Total Ownership Cost (TOC) and Schedule." SERC is a federally funded University Affiliated Research Center managed by Stevens Institute of Technology.

References

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery*, KDD '19, page 2623–2631, New York, NY, USA. Association for Computing Machinery.

Jean-Baptiste Alayrac, Adrià Recasens, Rosalia Schneider, Relja Arandjelović, Jason Ramapuram, Jeffrey De Fauw, Lucas Smaira, Sander Dieleman, and Andrew Zisserman. 2020. Self-Supervised MultiModal Versatile Networks. In *NeurIPS*.

AmirAli Bagher Zadeh, Paul Pu Liang, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2018. Multimodal language analysis in the wild: CMU-MOSEI dataset and interpretable dynamic fusion graph. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2236–2246, Melbourne, Australia. Association for Computational Linguistics.

Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. 2018. Multimodal machine learning: A survey and taxonomy. *IEEE transactions on pattern analysis and machine intelligence*, 41(2):423–443.

Tadas Baltrusaitis, Amir Zadeh, Yao Chong Lim, and Louis-Philippe Morency. 2018. Openface 2.0: Facial behavior analysis toolkit. In *2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018)*, pages 59–66. IEEE.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *arXiv:2004.05150*.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers.

Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Peter Hawkins, Jared Davis, Afroz Mohiuddin, Lukasz Kaiser, et al. 2020. Rethinking attention with performers. *arXiv preprint arXiv:2009.14794*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc Le, and Ruslan Salakhutdinov. 2019. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy. Association for Computational Linguistics.

Gilles Degottex, John Kane, Thomas Drugman, Tuomo Raitio, and Stefan Scherer. 2014. Covarep—a collaborative voice analysis repository for speech technologies. In *2014 ieee international conference on acoustics, speech and signal processing (icassp)*, pages 960–964. IEEE.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Alex Graves, Santiago Fernández, Faustino Gomez, and Jürgen Schmidhuber. 2006. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. In *Proceedings of the 23rd international conference on Machine learning*, pages 369–376.

Md Kamrul Hasan, Sangwu Lee, Wasifur Rahman, Amir Zadeh, Rada Mihalcea, Louis-Philippe Morency, and Ehsan Hoque. 2021. Humor knowledge enriched transformer for understanding multimodal humor. page online.

Md Kamrul Hasan, Wasifur Rahman, AmirAli Bagher Zadeh, Jianyuan Zhong, Md Iftekhar Tanveer, Louis-Philippe Morency, and Mohammed (Ehsan) Hoque. 2019. UR-FUNNY: A multimodal language dataset for understanding humor. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2046–2056, Hong Kong, China. Association for Computational Linguistics.

Devamanyu Hazarika, Roger Zimmermann, and Soujanya Poria. 2020. Misa: Modality-invariant and -specific representations for multimodal sentiment analysis. In *Proceedings of the 28th ACM International Conference on Multimedia*, MM '20, page 1122–1131, New York, NY, USA. Association for Computing Machinery.

Hao Hu and Guo-Jun Qi. 2017. State-frequency memory recurrent neural networks. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1568–1577, International Convention Centre, Sydney, Australia. PMLR.

Jian Huang, Jianhua Tao, Bin Liu, Zheng Lian, and Mingyue Niu. 2020. Multimodal transformer fusion for continuous emotion recognition. In *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3507–3511. IEEE.

Andrew Jaegle, Felix Gimeno, Andrew Brock, Andrew Zisserman, Oriol Vinyals, and Joao Carreira. 2021. Perceiver: General perception with iterative attention. *arXiv preprint arXiv:2103.03206*.

Urvashi Khandelwal, Kevin Clark, Dan Jurafsky, and Lukasz Kaiser. 2019. Sample efficient text summarization using a single pre-trained transformer. *arXiv preprint arXiv:1905.08836*.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. In *International Conference on Learning Representations*.

Jean Kossaifi, Robert Walecki, Yannis Panagakis, Jie Shen, Maximilian Schmitt, Fabien Ringeval, Jing Han, Vedhas Pandit, Antoine Toisoul, Bjoern W Schuller, et al. 2019. Sewa db: A rich database for audio-visual emotion and sentiment research in the wild. *IEEE transactions on pattern analysis and machine intelligence*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

G. Littlewort, J. Whitehill, T. Wu, I. Fasel, M. Frank, J. Movellan, and M. Bartlett. 2011. The computer expression recognition toolbox (cert). In *2011 IEEE International Conference on Automatic Face Gesture Recognition (FG)*, pages 298–305.

Zhun Liu, Ying Shen, Varun Bharadhwaj Lakshminarasimhan, Paul Pu Liang, AmirAli Bagher Zadeh, and Louis-Philippe Morency. 2018. Efficient low-rank multimodal fusion with modality-specific factors. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Melbourne, Australia. Association for Computational Linguistics.

Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. 2016. Hierarchical question-image co-attention for visual question answering. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, page 289–297, Red Hook, NY, USA. Curran Associates Inc.

Hongyuan Mei and Jason Eisner. 2017. The neural Hawkes process: A neurally self-modulating multivariate point process. In *Advances in Neural Information Processing Systems*, Long Beach.

Gábor Melis, Tomáš Kočiský, and Phil Blunsom. 2020. Mogrifier lstm. In *International Conference on Learning Representations*.

Daniel Neil, Michael Pfeiffer, and Shih-Chii Liu. 2016. Phased lstm: Accelerating recurrent network training for long or event-based sequences. *arXiv preprint arXiv:1610.09513*.

Zexu Pan, Zhaojie Luo, Jichen Yang, and Haizhou Li. 2020. Multi-Modal Attention for Speech Emotion Recognition. In *Proc. Interspeech 2020*, pages 364–368.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Hai Pham, Paul Pu Liang, Thomas Manzini, Louis-Philippe Morency, and Barnabás Póczos. 2019. Found in translation: Learning robust joint representations by cyclic translations between modalities. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6892–6899.

Jack W. Rae, Anna Potapenko, Siddhant M. Jayakumar, Chloe Hillier, and Timothy P. Lillicrap. 2020. Compressive transformers for long-range sequence modelling. In *International Conference on Learning Representations*.

Wasifur Rahman, Md Kamrul Hasan, Sangwu Lee, AmirAli Bagher Zadeh, Chengfeng Mao, Louis-Philippe Morency, and Ehsan Hoque. 2020. Integrating multimodal information in large pretrained transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online. Association for Computational Linguistics.

Zhongkai Sun, Prathusha Sarma, William Sethares, and Yingyu Liang. 2020. Learning relationships between text, audio, and video via deep canonical correlation for multimodal language analysis. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8992–8999.

Yi Tay, Dara Bahri, Donald Metzler, Da-Cheng Juan, Zhe Zhao, and Che Zheng. 2020. Synthesizer: Rethinking self-attention in transformer models. *arXiv preprint arXiv:2005.00743*.

Yao-Hung Hubert Tsai, Shaojie Bai, Paul Pu Liang, J. Zico Kolter, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2019. Multimodal transformer for unaligned multimodal language sequences. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Florence, Italy. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, undefinedukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 6000–6010, Red Hook, NY, USA. Curran Associates Inc.

Yansen Wang, Ying Shen, Zhun Liu, Paul Pu Liang, Amir Zadeh, and Louis-Philippe Morency. 2019. Words can shift: Dynamically adjusting word representations using nonverbal behaviors. *Proceedings of the AAAI Conference on Artificial Intelligence*, 33(01):7216–7223.

Martin Wöllmer, Felix Weninger, Tobias Knaup, Björn Schuller, Congkai Sun, Kenji Sagae, and Louis-Philippe Morency. 2013. Youtube movie reviews: Sentiment analysis in an audio-visual context. *IEEE Intelligent Systems*, 28(3):46–53.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Amir Zadeh, Minghai Chen, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2017. Tensor fusion network for multimodal sentiment analysis. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark. Association for Computational Linguistics.

Amir Zadeh, Paul Pu Liang, Navonil Mazumder, Soujanya Poria, Erik Cambria, and Louis-Philippe Morency. 2018. Memory fusion network for multiview sequential learning. *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*.

Amir Zadeh, Rowan Zellers, Eli Pincus, and Louis-Philippe Morency. 2016. Mosi: multimodal corpus of sentiment intensity and subjectivity analysis in online opinion videos. *arXiv preprint arXiv:1606.06259*.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems*, volume 33, pages 17283–17297. Curran Associates, Inc.

Haoyi Zhou, Shanghang Zhang, Jieqi Peng, Shuai Zhang, Jianxin Li, Hui Xiong, and Wancai Zhang. 2021. Informer: Beyond efficient transformer for long sequence time-series forecasting. In *The Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021*, page online. AAAI Press.

A Statistics of Datasets

Table 4: Statistics for pre-processed CMU-MOSI and CMU-MOSEI datasets from Tsai et al. (2019). For each column, "A" represents aligned, "UA" represents unaligned. "A", "T", "V" in each row respectively represents average sequence lengths for these modalities. "Train", "Test", "Valid" represent the number of data points for each split. The sequences are pre-truncated and padded which makes all samples from one modality have the same length.

	MOSI		**MOSEI**	
	A	UA	A	UA
A	50	375	50	500
T	50	50	50	50
V	50	500	50	500
Train	1284		16265	
Test	686		4643	
Valid	229		1869	

Table 5: Statistics for the UR-FUNNY dataset, “target” denote the average length of the target sentence used for classification, “context” denotes the average length of the preceding utterances. The audio, video and text data in the official pre-processed UR-FUNNY dataset are pre-aligned. We concatenate the context and target for prediction.

	Train	Test	Valid
Avg. target length	15.81	16.55	16.94
Avg. context length	41.69	42.86	43.94
Avg. num contexts	2.84	2.95	2.81
Num	10598	3290	2626

B Hyper-parameter Optimization

Table 6: Settings for hyper-parameter optimization.

	Range	Step size	Distribution
lr	[5e-5, 2e-3]	-	Log uniform
Layers	[2, 8]	2	Uniform
S	[2, 8]	1	Uniform
r	[1, 8]	1	Uniform
Dropout	[0, 0.3]	0.05	Uniform
α	[0, 3]	1	Uniform
β	[0, 0.5]	-	Uniform
γ	[0, 5]	0.5	Uniform

We use Optuna hyperparameter optimization framework (Akiba et al., 2019) to perform grid search on hyper-parameters. Optimized hyper-parameters, search space, and distributions are listed in Table 6. r, α, β, γ for the three sub-layers are optimized independently with the same setting. Dropout rates for the attention layer, FFNs, and embedding layer are optimized independently with the same setting.

Attention head number is fixed to 8. d_{model} is fixed to 32. The number of epochs for CMU-MOSI, CMU-MOSEI, UR-FUNNY are 100, 50, 100 respectively. We use the maximum batch size that can fit on a single NVIDIA Tesla V100 memory. Gradient clipping is done for norms of 0.8, 1.0, 1.0 for CMU-MOSI, CMU-MOSEI, UR-FUNNY respectively. We use Adam for optimization with the default hyper-parameters from PyTorch. We use a plateau learning rate scheduler that decreases the learning rate by a factor of 10 when the validation performance plateaus and with a patience of 20 epochs.

C SPT that use concatenation

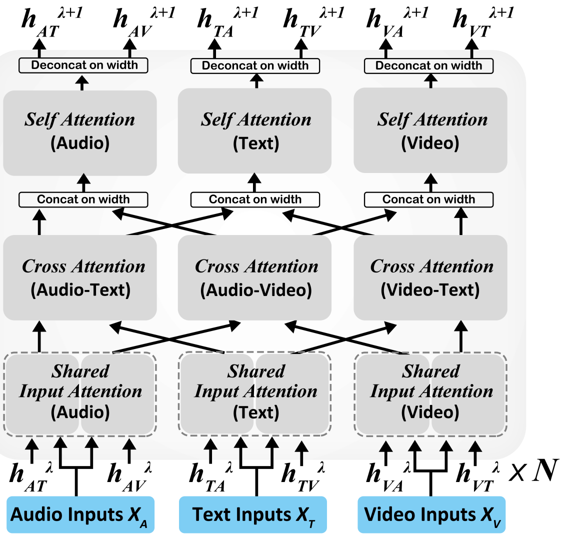

Figure 6: SPT using concatenation instead of summation.

D SPT that implement serial structure

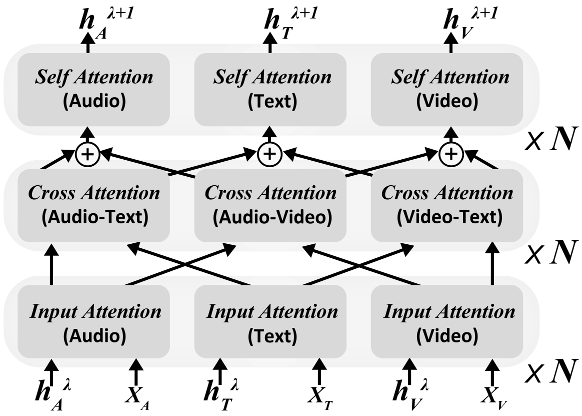

Figure 7: SPT using serial structure.