S-Omninet: Structured Data Enhanced Universal Multimodal Learning Architecture

Ye Xue[1] , **Diego Klabjan**[2] and **Jean Utke**[3]
[1,2]Northwestern Univerison
[3]Allstate Insurance Company
yexue2015@u.northwestern.edu, d-klabjan@northwestern.edu, jutke@allstate.com

Abstract

Multimodal multitask learning has attracted an increasing interest in recent years. Singlemodal models have been advancing rapidly and have achieved astonishing results on various tasks across multiple domains. Multimodal learning offers opportunities for further improvements by integrating data from multiple modalities. Many methods are proposed to learn on a specific type of multimodal data, such as vision and language data. A few of them are designed to handle several modalities and tasks at a time. In this work, we extend and improve Omninet, an architecture that is capable of handling multiple modalities and tasks at a time, by introducing cross-cache attention, integrating patch embeddings for vision inputs, and supporting structured data. The proposed Structured-data-enhanced Omninet (S-Omninet) is a universal model that is capable of learning from structured data of various dimensions effectively with unstructured data through cross-cache attention, which enables interactions among spatial, temporal, and structured features. We also enhance spatial representations in a spatial cache with patch embeddings. We evaluate the proposed model on several multimodal datasets and demonstrate a significant improvement over the baseline, Omninet.

1 Introduction

Deep learning yielded great success in the last decade on tasks across many domains with various types of unstructured data such as images and text. Specific models are carefully designed and tailored for a particular type of data. For example, image recognition models [He *et al.*, 2016] are built with convolutional neural networks (CNNs), while recurrent neural network (RNN) has shown success for sequential data. Although some recent works use CNN in language tasks and transformer on image tasks [Dosovitskiy *et al.*, 2021], these tasks are still single-modal tasks.

In recent years, multimodal learning attracted an increasing interest, as the single modal models are advancing rapidly and achieving astonishing results on various tasks across multiple domains, such as vision [Dosovitskiy *et al.*, 2021] and language [Devlin *et al.*, 2019]. Multimodal tasks are more common in complex tasks and multimodal learning has been shown more effective than models that used only single modalities in several fields, such as healthcare, where models are trained to utilize medical images and electronic health records [Huang *et al.*, 2020; Li *et al.*, 2021], and autonomous driving where intelligent systems are built to process various signals in different modalities [Feng *et al.*, 2020; Prakash *et al.*, 2021].

However, existing multimodal learning models are still usually tailored to specific tasks or modalities and are not easily extended to accommodate new types of data. Different models for different tasks need to be separately designed and maintained. In business, new tasks may be added over time as a business grows and new types of data may also need to be considered to best utilize the extra information. One architecture that handles multiple modalities and multiple tasks becomes increasingly appealing.

MultiModel [Kaiser *et al.*, 2017] is probably the first attempt at building a single model that can solve tasks across multiple domains. It consists of several modality nets, an encoder, and a decoder. Each modality net handles one type of input data. A mixer module gathers encodings from modality nets and feeds them to the decoder. However, for a single task, MultiModel does not have support for inputs having more than one modality, such as Visual Question Answering (VQA). In order to address this challenge, another multi-model multi-task architecture, Omninet, is proposed [Pramanik *et al.*, 2019]. Similar to modality-nets, Omninet has a visual peripheral to encode images and videos and several language peripherals to encode text data in different languages. Inputs in different modalities are further encoded into temporal and spatial caches, which are fed into a transformer-based decoder.

Omninet is shown to have competitive performance in several tasks compared with state-of-the-art models. However, there are still a few shortcomings of Omninet. First, each modality in Omninet is encoded in a separate stream. Existing works [Tsai *et al.*, 2019; Tan and Bansal, 2019; Lu *et al.*, 2019] have shown that it is beneficial for multimodal learning models to encode one modality with the information of other modalities. Second, it lacks support for structured data. In many real world applications structured data play an important role, even in a multimodal scenario.

For example, similar medical image findings may suggest different diagnoses given different laboratory test results [Huang *et al.*, 2020]. Unfortunately, many of the practical applications are built on proprietary data sets making this specific topic less accessible for academic research. However, there are countless cases where full context, in the form of structured and unstructured, is critical for making accurate decisions. A straightforward way of extending Omninet to utilize structured data is concatenating structured features with the final encoding vector of the other modalities. However, such a late fusion mechanism ignores the potential informative interactions between structured and unstructured data. Furthermore, it would not deal with a varying number of structured data sources, because naive concatenation fixes the number of structured data sources in the network setup.

In this work, we extend Omninet with a design of a structured peripheral and structured cache. Instead of common encoding methods that encode categorical structured features into one vector, such as one-hot encoding, we encode structured data using entity embeddings [Guo and Berkhahn, 2016]. We store encodings of structured data into the structured cache. It interacts with other caches through a cross cache attention mechanism, which we propose to enhance the encodings by considering those from other caches. We also modify the image peripheral to produce lower-level representations and divide the encoded images into patches before interacting with other caches. High-level representations used in Omninet might lose spatial signals which can be very useful to help encode other caches.

Our main contributions are summarized as follows.

1. We extend Omninet to handle structured data effectively and to deal with a various number of structured data sources.
2. We enhance its encoding process with cross cache attention and incorporate the idea of patches to enable cross cache interactions on lower level image representations.
3. The proposed model is evaluated on several multimodal datasets, which cover a wide range of modalities, including images, textual inputs, structured data, and videos. It demonstrates a significant improvement against Omninet on all datasets.

The source code is will be disclosed once the paper is accepted.

In Section 2, we discuss related work. The proposed model is described in Section 3. The datasets and experimental setup are described in Section 4. Section 5 discusses the computational results and the conclusions are drawn in Section 6.

2 Related Work

Most existing works in multimodal learning focus on specific tasks with a fixed set of modalities, such as images and text. Many works concatenate image and text inputs and encode them together with Transformer [Chen *et al.*, 2020; Li *et al.*, 2020]. VideoBERT [Sun *et al.*, 2019] and VisualBERT [Li *et al.*, 2019] extend such a transformer-based model to video data. ViLBERT [Lu *et al.*, 2019] keeps a separate stream for each modality and enables cross-modality connections to encode one modality with the other.

Another category of works extends a multimodal learning model in the context of multiple tasks. With the encoder-decoder architecture, MultiModel [Kaiser *et al.*, 2017] and UniT [Hu and Singh, 2021] are able to handle multiple tasks, such as classification and sequence prediction, with just one model. For a single task, MultiModel does not have support for inputs having more than one modality. UniT addresses this issue by encoding images and text with transformers and concatenating encodings from both modalities. It is still limited to only image and language modalities. Perceiver IO [Jaegle *et al.*, 2021] is able to handle tasks with different modalities. However, it does not support multiple tasks of various modalities at the same time. To the best of our knowledge, Omninet [Pramanik *et al.*, 2019] is the most general architecture for multimodal and multi-task learning. Multiple tasks with inputs in different modalities can be trained together in one model. However, it lacks support for structured data. In addition, the lack of interactions between caches limits its ability in the encoding process.

To accommodate structured data, Omninet can be easily extended using late fusion, which is widely used in combining structured data with other features [Liu and El-Gohary, 2020; Zhang *et al.*, 2020b]. For example, we can encode structured data in one-hot encoding with a few linear layers and concatenate the output with the unstructured feature vector. However, it has a few limitations. The dimension of concatenated structured data depends on the number of structured data sources, which means the model's dimension needs to be changed to adapt to new use cases when the number of structured data sources are different. In addition, the information in the structured data may help in encoding other modalities but late fusion mechanisms do not provide such interactions. Entity embeddings [Guo and Berkhahn, 2016] have been used to encode structured data [Zhu *et al.*, 2020; Kulkarni *et al.*, 2021]. The encoded structured features are usually concatenated into one feature vector and then combined with other features using late fusion techniques [Kulkarni *et al.*, 2021].

Our proposed cross-cache attention mechanism falls into the category of attention-based fusion [Zhang *et al.*, 2020a]. Previous works using attention-based fusion in multimodal learning focus on vision-language interactions [Tsai *et al.*, 2019]. This kind of attention-based fusion has not yet been studied on structured data and image patches in the multimodal multitask learning scenario.

3 Model

Figure 1 shows the architecture of the proposed model. A single sample input can be any combination of zero or more of the following modalities: an image $X_{image} \in \mathbb{R}^{H \times W \times C}$, video frames $X_{video} \in \mathbb{R}^{F \times H \times W \times C}$, textual input (i.e., sentences of Q words) and structured data $X_{structured} \in \mathbb{R}^{M}$. For example, a sample can be one or more images, one or more sentences and a video. The number of video frames F, length of textual input Q and the number of structured features M can be different across samples.

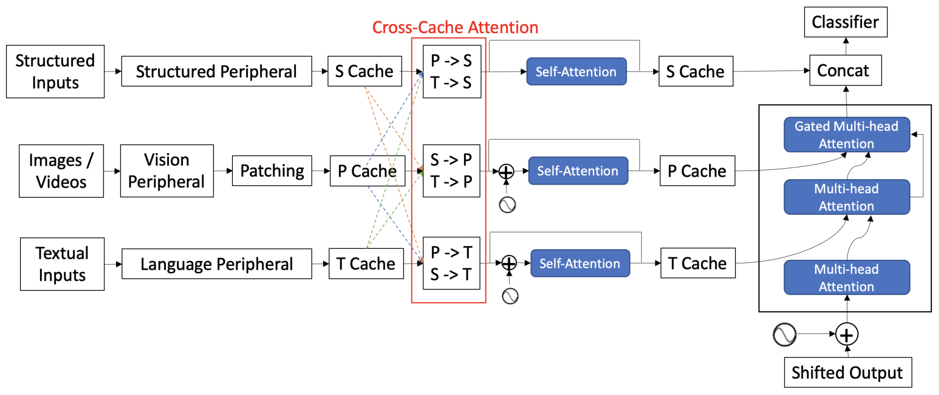


Figure 1: S-Omninet architecture. Caches are denoted as structured (S) cache, spatial (P) cache and temporal (T) cache.

Omninet has a peripheral for each different modality. A peripheral produces intermediate encodings of the corresponding modality. There are 2 lists. Modalities of spatial matrix are added to the spatial and those corresponding to sequences to the temporal cache list. A sequence of spatial matrices (e.g., video frames) has both spatial and temporal aspects. Omninet encodes each spatial matrix through the spatial peripheral and also stores encodings in the spatial cache. Each matrix encoding is also aggregated through a pooling layer and stored in the temporal cache. A self-attention based temporal encoder is used to calculate embeddings of the temporal cache. Both caches are fed to the Decoder, which is based on the decoder architecture from Transformer [Vaswani *et al.*, 2017] with an additional gated-attention layer to handle the spatial cache. There is also a domain/task encoding. The domain encoding is appended to intermediate encodings from a peripheral to distinguish among different modalities. The task encoding is used as the input of the Decoder to identify different tasks.

There are 3 aspects that Omninet does not capture: (1) structured data are neither spatial nor temporal and thus cannot be directly modeled; (2) in the embedding phase the two caches are treated independently of each other while often there is interaction, and (3) the spatial cache is used at the pixel level with no notion of locality.

In the following subsections, we introduce in detail the encoder of S-Omninet, including the new structured stream (structured peripheral and structured cache), the enhancement of spatial cache through patching, the cross-cache attention modules and additional self-attention modules on caches other than temporal cache.

3.1 Structured Peripheral and Cache

We propose to use a structured peripheral to encode structured data and put them into the structured cache. In order for the structured cache to effectively communicate with the other two caches, we use entity embeddings to encode categorical features in the structured peripheral. Let us assume there are C categorical features, denoted as $s_1, s_2, ..., s_C$. Each state of a categorical feature is mapped to a vector as $s_i \rightarrow \mathbf{s}_i \in \mathbb{R}^D$ through a trainable embedding layer of dimension D. It functions as if we "tokenize" all possible states of each feature. For example, a color feature can have different embedding vectors for the value 'red' and 'blue.' Our model may learn similar embeddings for structured value 'red' and textual input 'red.' We argue that this helps the model match similar concepts between structured and unstructured data.

As the entity embeddings encode each category separately, we may lose useful patterns that can be learned from all structured features as a whole. Therefore, we also keep the encoding of the whole structured data including continuous features besides entity embeddings. The structured peripheral transforms the whole structured sample $X_{structured}$ using one-hot encodings and encodes them through linear layers. The encoded features are denoted as $\mathbf{s} \in \mathbb{R}^D$. We then append structured domain encoding to entity embedding features and the traditional whole structured feature and project them back to the dimension D before inserting them into the structured cache. Multiple structured data sources can be encoded and handled by the structured cache. We denote the structured cache containing N_s encodings as $X_s \in \mathbb{R}^{N_s \times D}$.

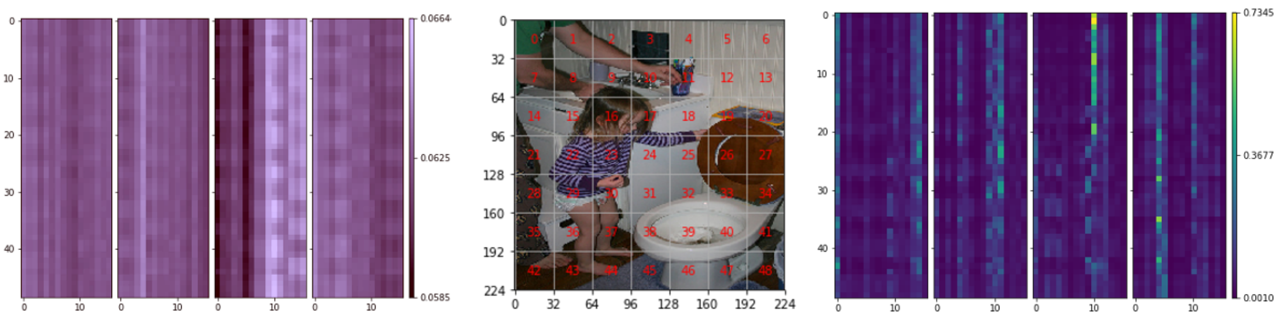
The structured cache is further encoded in the cross-cache attention modules and self-attention layers, which we introduce later. We use the first encoding of the output from self-attention layers as a representation of the whole structured data and concatenate it with the decoder output before the final prediction.

3.2 Spatial Cache

Patches

Spatial components of a sample are encoded by a vision peripheral, which produces an $H' \times W' \times d_m$ feature map for an image and F such feature maps for a video, for example. Instead of directly flattening the feature maps and storing them in the spatial cache as done by Omninet, we divide the feature maps into a sequence of 2D patches, similar to ViT [Dosovitskiy *et al.*, 2021]. Compared with Omninet, the patches in our model preserve more spatial information than the highly encoded feature maps.

A patch of a matrix feature map at location (i, j) with height p_h and width p_w is the rectangle area of the matrix with two diagonal coordinates (i, j) and $(i+p_h, j+p_w)$. We obtain patches at each valid location of a matrix feature map with a certain stride. A location is valid if we can obtain a



(a) Attentions with early self-attention (b) VQA image (c) Attentions with late self-attention

Figure 2: An example of VQA image (b) and the attention maps (a, c) in the cross-cache attention $CCA(X_p, X_t)$. In (a) and (c), we plot 4 attention maps, one for each head in the attention. The spatial cache is on the x-axis and the temporal cache is on the y-axis. The question for this image is "Is the girl potty-trained?" The encoding of "girl" is indexed at 9. The word "potty" is encoded with two subwords, "pot" and "ty," which are indexed at 10 and 11.

patch without stepping out the boundary. We denote a sequence of patches of a matrix feature map as $\mathbf{z}_p \in \mathbb{R}^{T_p \times d_p}$, where $T_p = (H'W')/(p_h p_w)$ is the number of patches and $d_p = (p_h \cdot p_w) \cdot d_m$. Each patch is mapped to the desired dimension D that matches the size of latent vectors in our attention layers in the spatial cache. All encoded patches of a matrix feature map are denoted as $\mathbf{x}_p^0 = [\mathbf{z}_p^1\mathbf{E}, \mathbf{z}_p^2\mathbf{E}, ..., \mathbf{z}_p^{T_p}\mathbf{E}]$, where $\mathbf{E} \in \mathbb{R}^{d_p \times D}$. For sequences of matrices (videos), we divide each matrix's feature map into patches and store patches of all matrices in the spatial cache.

Positional Embeddings

A position embedding is added to each patch to retain position information. A sequence of encoded patches is denoted as $\mathbf{x}_p = \mathbf{x}_p^0 + \mathbf{E}_{pos}$, where $\mathbf{E}_{pos} \in \mathbb{R}^{T_p \times D}$. We use learnable positional embeddings as it shows a better performance than the fixed positional embeddings in ViT [Dosovitskiy *et al.*, 2021].

In the model, the patches may also come from videos, which means temporal relations may exist among patches. In order to capture both the spatial and temporal aspects, two sets of embeddings are learned, each for one of the aspects and each with size $T_p \times \frac{D}{2}$. The spatial embedding $\mathbf{E}_{pos}^p \in \mathbb{R}^{T_p \times \frac{D}{2}}$ retains a patch's position information within an image or a video frame. We encode 2D positional information in the spatial embedding and two sets of embeddings are learned, one for each axis. Specifically, we learn X-embedding $\mathbf{E}_{posx}^p \in \mathbb{R}^{T_p \times \frac{D}{4}}$ and Y-embedding $\mathbf{E}_{posy}^p \in \mathbb{R}^{T_p \times \frac{D}{4}}$. Based on the patch's coordinates, we concatenate the X-embedding and Y-embedding to obtain its spatial embedding. The temporal embedding $\mathbf{E}_{pos}^f \in \mathbb{R}^{F \times \frac{D}{2}}$ captures the position of the frame. Then, based on a patch's index of the frame and its position inside the frame we concatenate $\mathbf{E}_{pos}^f$ and $\mathbf{E}_{pos}^p$ to get the final temporal-spatial embedding $\mathbf{E}_{pos}$.

Patches are further encoded with the corresponding domain encoding before added to the spatial cache. We denote the spatial cache with a total of N_p encoded patches as $X_p \in \mathbb{R}^{N_p \times D}$.

3.3 Temporal Cache

The temporal cache consists of encodings from domains that have a temporal dimension, such as textual inputs and videos. Textual inputs are encoded first by the corresponding language peripheral and domain encoding. Then they are further encoded by a temporal encoder before put in the temporal cache. For videos, the temporal cache stores frame-level features, which are obtained through pooling from patches. We denote the temporal cache with N_t encodings as $X_t \in \mathbb{R}^{N_t \times D}$.

3.4 Cross-cache Attention

We define the cross-cache attention of cache $X_\alpha \in \mathbb{R}^{T_\alpha \times d_\alpha}$ and $X_\beta \in \mathbb{R}^{T_\beta \times d_\beta}$ as $CCA(X_\alpha, X_\beta) \in \mathbb{R}^{T_\alpha \times D}$. The destination cache X_α provides queries and the source cache X_β provides keys and values. Our model consists of 3 streams of cross-cache attention:

$$\begin{aligned} Y_s &:= concat(CCA(X_s, X_p), CCA(X_s, X_t)), \\ Y_t &:= concat(CCA(X_t, X_p), CCA(X_t, X_s)), \\ Y_p &:= concat(CCA(X_p, X_s), CCA(X_p, X_t)). \end{aligned}$$

Different from other cross-modality attention models [Tsai *et al.*, 2019; Tan and Bansal, 2019], our model consists of a stream for the structured modality. Unstructured inputs are broken down into spatial and temporal caches instead of encodings of each modality separately. Additionally, instead of performing cross-modality attention on the pixel-level embeddings, our model captures spatial information during cross-cache attention by taking advantage of the patch features.

3.5 Late Self Attention

As shown in Figure 1, we add self-attention layers after cross-cache attentions. In the original design of Omninet, modalities with the temporal dimension are encoded with self-attention based temporal encoder before being put into caches. We initially applied cross-cache attentions after the self-attention layers. However, we found that cross-cache attentions cannot capture interactions effectively. As shown in

Figure 2a, we see that all spatial cache encodings have almost the same attention pattern on temporal cache encodings. The attention scores are all very close, ranging from 0.058 to 0.067. The attention scores on each row sum up to 1. For each spatial cache encoding, its attention scores on all 16 temporal cache encodings are almost the same, close to the average $1/16$ or 0.0625. The reason is that self-attention makes the encodings in temporal cache similar, i.e., having large cosine similarity. By moving the self-attention layers after the cross-cache attentions, we observe a different pattern, as shown in Figure 2c. It shows that cross-cache attentions pay various attentions to different spatial-temporal encoding pairs. The most relevant pair gets an attention score around 0.7, while the scores of irrelevant pairs stay well below 0.3. One encoding getting an attention score of 0.7 means the other 15 encodings together get only 0.3. The attention scores vary much larger than the previous case, see more discussions in Section 5.1. Note that, MulT [Tsai *et al.*, 2019] also put self-attention layers after their cross-modality attentions. The authors empirically argue that this design benefits cross-modality attention without further explanations.

In Omninet, the whole frame encodings are also stored in the temporal cache, so it can learn temporal correlations between frames. We preserve this design in our model. However, this prevents cross-cache attention from learning cross-cache correlations effectively. The reason is that the patch encodings are much closer, in terms of cosine similarity, to the frame encodings than other temporal cache encodings coming from other modalities. This results in high attention scores on the frame encodings, which overshadow the interactions between spatial cache and other temporal cache encodings. Therefore, we exclude the frame encodings in the cross-cache attention.

3.6 Residual Connections

Residual connections are commonly used in transformers [Vaswani *et al.*, 2017; Dosovitskiy *et al.*, 2021]. We also add residual connections after self-attention blocks to mitigate the vanishing gradient problem as our network is even deeper than Omninet. Additionally, since we put self-attention blocks after cross-cache attentions, the model may lose some cross-cache signals learned previously, which are critical to the predictions in many cases. The residual connections keep the cross-cache signals and merge them with the outputs of self-attention blocks. We observe a significant improvement by adding residual connections. We also verify that the residual connections are active in inference by comparing the weights between the residual connections and the outputs of the self-attention blocks.

3.7 Universal Architecture and Configurations

S-Omninet is a universal architecture, i.e., we have one and only one model for all tasks. The configuration, such as the sizes of dense layers and the number of attention layers, is also the same across tasks. The dimension of embeddings D in both encoder and decoder is 512. The vision peripheral produces 14×14 feature maps and the patch size is 2×2. The self-attention blocks on the temporal cache have 6 layers and 8 heads, the same as Omninet. In other attention blocks, including self-attention blocks on other caches and all cross-cache attention blocks, we use 3 layers and 4 heads.

Vision and language peripherals are pre-trained and fixed during training of S-Omninet. We train the structured peripheral along with the main architecture of the model. The reason is that different structured data can have very different patterns and it does not make much sense to use a structured peripheral that is pre-trained on totally different features. In addition, the structured peripheral is light and relatively easy to train along with the main model.

For classification tasks, we add dense layers after the decoder as the prediction head. For frame generation tasks, we use the generator model in DCGAN [Radford *et al.*, 2016] and modify the configurations to fit our feature and frame dimensions. Excluding the pre-trained peripherals and prediction heads, S-Omninet and Omninet has 125.4 million and 96.6 million trainable parameters, respectively.

Table 1: Test performance comparison

Datasets	Omninet	S-Omninet	Improvement(%)
VQA	56.3	57.3	1.83
S-VQA	61.1	61.7	1.01
Social-IQ	64.7	66.9	3.31
MOSI-Sen	75.5	78.6	4.22
MOSI-Gen	0.116	0.013	2.73

4 Datasets

Social-IQ Social-IQ [Zadeh *et al.*, 2019] is a video question answering dataset that contains 1,250 annotated videos, 7,500 questions and 52,500 answers. Each question is provided with 4 correct answers and 3 incorrect answers. All answers are sentences. The task is to predict whether an answer is correct given a video with a question. We extract video frames at 1fps [Zadeh *et al.*, 2019] and each video sample consists of 55 frames on average.

CMU-MOSI CMU-MOSI [Zadeh *et al.*, 2016] is a multimodal human sentiment dataset. It consists of 2,199 video clips of faces during conversations. Each video clip is labeled with a sentiment score between -3 and 3. We evaluate the model performance on two tasks. One is MOSI-Sen, a binary classification task to predict whether the sentiment is positive or negative using video frames and transcripts. The transcripts, text translated from video's audio, are provided in the dataset. The other task is MOSI-Gen where we generate the next frame of a video clip, given previous frames and the transcript of this clip. We use accuracy for the classification task and Mean Absolute Error (MAE) for the frame generation task.

Visual Question Answering The VQA v2.0 dataset [Goyal *et al.*, 2017] is a large visual question-answering dataset consisting of approximately 1.1 million (image, question) pairs with 13 million answers on MSCOCO images. Every question is associated with two similar images that result in two different answers. We evaluate the model performance on the provided test-dev set same as Omninet [Pramanik *et al.*, 2019].

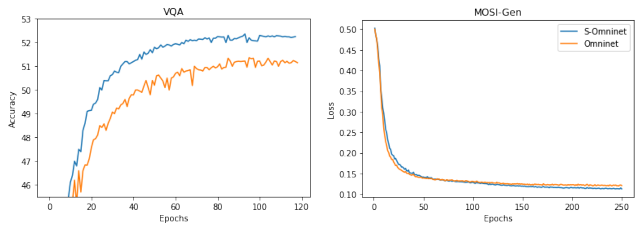


Figure 3: Learning curves on the validation set

Structured & Visual Question Answering Due to the lack of a public multimodal dataset that contains structured data and multiple unstructured data, we create the S-VQA dataset which contains images, text and structured data. In practical business processes, this type of data is very often encountered which motivates this study. A single model for a given process serves different specific use cases. Unfortunately, the lack of established multi-modal approaches, combined with such data sets being proprietary means no well-curated data sets are available. Therefore, we create the S-VQA that contains a large number of multimodal samples.

The key part is to create structured data that have a meaningful interaction with other modalities to simulate the real-world cases. The structured data are composed of both numerical and categorical features. We create it in such a way that the numerical features are correlated with the labels and categorical features are correlated with both the labels and the unstructured data. For numerical features, we randomly sample n_c vectors with the desired dimension and treat them as the *centroids*. The value of n_c is the same as the number of classes in the VQA v2.0 dataset. Given a covariance matrix, we sample random vectors around each *centroid* from Gaussian distributions. Vectors sampled from the same *centroid* are assigned with the same class label. A total of 3,500 clusters are generated to be matched with all classes. For categorical features, we first identify important elements (words in a sentence or regions of an image) in existing modalities. The importance of each element is determined by the attention score produced by the decoder of the vanilla Omninet during training. Then, we create categorical features by clustering the important elements. Each cluster is considered a categorical feature. In addition, we perturb the generated structured features to introduce correlations to the labels. More details are in Appendix A.

5 Results

We build S-Omninet on top of Omninet by adding implementations of the cross-cache attention module, the patching module and the structured peripheral. For the vision peripheral, we use a pre-trained ResNet-152 model [He *et al.*, 2016] with the last pooling layer and a few convolution layers removed to get the desired 14×14 feature maps. We use the same language peripheral as Omninet. We run experiments on NVIDIA GeForce RTX 2080 Ti GPUs.

Table 1 shows the performance comparison between our model and the baseline. On the VQA and S-VQA datasets, we observe an improvement of 1.83% and 1.01% on the accuracy, respectively. On the social-IQ dataset, our model achieves an accuracy of 66.9, which is 3.31% better than Omninet. The accuracy scores of both our model and Omninet are higher than 63.91, which is a baseline performance as reported in the original paper [Zadeh *et al.*, 2019]. On MOSI-Sen and MOSI-Gen, S-Omninet is 4.22% better in accuracy on the sentiment prediction task and 2.73% better in MAE on the frame prediction task.

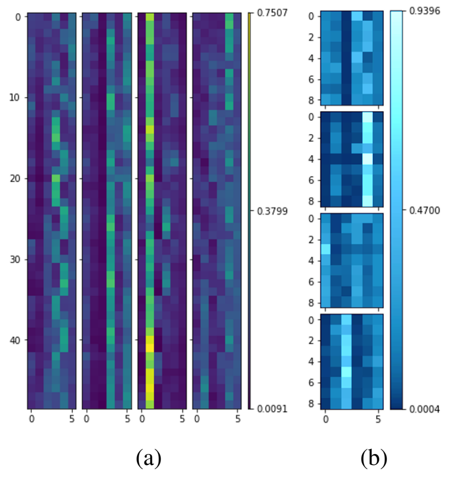

(a) (b)

Figure 4: Cross-cache attention on structured features. The structured cache is on the x-axis and the other cache is on the y-axis. Left: attention maps in $CCA(X_p, X_s)$. Right: attention maps in $CCA(X_t, X_s)$.

Figure 3 shows the validation accuracy curves on VQA and validation loss curves on MOSI-Gen. S-Omninet has a similar converging behavior as Omninet. The curves on the other datasets demonstrate a similar pattern. However, since our model is slightly larger than Omninet, the actual training time is 23% longer per epoch than Omninet.

5.1 VQA

As shown in Table 1, our model shows significant improvements over Omninet. Omninet's performance on the VQA dataset was first reported as 55.3 [Pramanik *et al.*, 2019]. With more tuning, we achieve an accuracy of 56.3 and use this model as the baseline. Our model improves the baseline by 1.83%, which demonstrates the benefits of adding cross-cache attention.

We also test our model in a more challenging task, "two-question VQA," which demonstrates the effectiveness of cross-cache attention more clearly. In this task, we randomly add an irrelevant question before or after the original question. For example, the textual input of the example in Figure 2 is "What is this a collection of? Is the girl potty-trained?" The second sentence is the original question for this image and the first sentence is an irrelevant question.

On this "two-question" VQA dataset, we argue that Omninet cannot tell which question is the relevant one due to the lack of interactions between caches. Since each cache is encoded separately, Omninet cannot use spatial information to identify the relevant question. Therefore, the performance of Omninet is substantially impacted and the accuracy drops

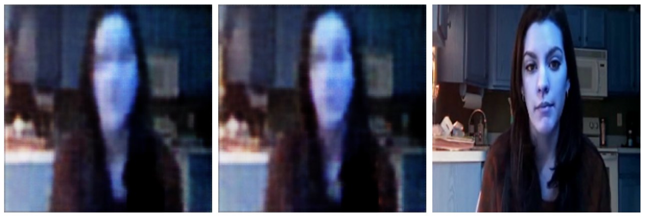


Figure 5: Generated frames from different models

from 51% to 41%. In contrast, our model with cross-cache attention shows a good ability to identify the relevant question. Figure 2b shows an example of a VQA input and Figure 2c shows the attention maps in cross-cache attention on this example. In the cross-cache attention $CCA(X_p, X_t)$, image patches are encoded with the textual features. As shown in Figure 2c, words in the irrelevant question, indexed from 0 to 8 on the x-axis, get low attention scores. As a result, our model shows an 8% improvement in accuracy over Omninet on this challenging task.

Besides paying more attention to the correct sentence, cross-cache attention also stresses more relevance to more relevant words. We can see that in all 4 heads, the words "girl" and "potty" have a high attention score in many cases. Especially in the second head, the word "girl" and "potty" are linked with the regions that show the girl and potty. Note that, we mark grids in the original image to ease the visualization, but the grid indices are not strictly mapped to patch indices on the y-axis. The reason is that the patch embeddings used in cross-cache attention come from CNN-encoded feature maps. Each patch also contains information in its surrounding patches due to the convolution operations.

5.2 Cross-cache Attention on Structured Data

Omninet does not handle structured data. In order to train Omninet on S-VQA, we encode the structured data with one-hot encodings followed by a linear layer. Then the encoded structured data is concatenated with the output of the decoder. On the S-VQA dataset, Omninet achieves an accuracy of 61.1%, higher than the accuracy on the VQA dataset, as the structured data provides extra information about the labels. Our model further improves Omninet by 1.01% relatively. In the structured cache, the first encoding is the embedding of the whole structured feature vector including numerical and categorical features. The second encoding is the embedding of categorical features generated from important spatial input signals. The rest are encodings of categorical features of important temporal inputs. Figure 4 shows the attention maps on cross-cache attention $CCA(X_p, X_s)$ and $CCA(X_t, X_s)$, where the model encodes each cache with more attention on corresponding structured encodings. For example, the spatial cache is encoded with more attention on the second structured encoding and the temporal cache is encoded with more attention on the third and fifth structured encodings. It demonstrates the effectiveness of cross-cache attention in integrating correlated structured and unstructured data.

5.3 Video Tasks

Social-IQ and CMU-MOSI are both video datasets with different tasks. On classification tasks, we observe significant improvements against Omninet. Our model is 3.3% and 4.2% better than Omninet on Social-IQ and MOSI-Sen, respectively. The cross-cache attention modules enable caches to interact with each other, which helps the model locate related encodings in different caches as we see in the "two-question VQA" example. This module can be more useful on videos than images because there are multiple frames in a video clip and the model needs to figure out in which frame encodings are more relevant.

Although Omninet has the Gated Multihead Attention, which makes the model focus more on important frames by increasing the attention scores on them, there are several limits to this mechanism. First, the importance of a frame is calculated from an attention module on the temporal cache, which includes the frame and word embeddings. The frame-level embedding is a highly encoded image feature, where detailed spatial information, which can be critical to deciding whether a frame is important, is missing. Second, once a frame is identified as an important frame, all encodings in this frame get a higher weight in the next attention. Because of that, a less important encoding in an important frame can have a higher weight than an important encoding in a less important frame. The cross-cache attention overcomes these problems by performing attentions in a finer granularity, i.e., directly on the encoding level.

On the frame generation task, our model shows a 2% improvement on MAE and we observe that our model generates more eye appealing images, as shown in Figure 5. The image generated by our model is darker on the eyes, nose and mouth, which shows clearer boundaries of facial features. Both frames that are generated by Omninet and our model are not very crisp. This is often seen on transformer-based models with a simple image generator [Jaegle *et al.*, 2021] (this work's focus is not on generating high resolution and sharp images). In spite of that, we still see that the image from our model has additional facial details than Omninet.

6 Conclusion

In this work, we extend and improve Omninet by introducing cross-cache attention, integrating patch embeddings for vision inputs, and supporting structured data. We discuss the design choice of putting cross-cache attention before self-attentions. In addition, we study the impact of this design and demonstrate reasons it works. The proposed S-Omninet is shown capable of learning structured data of various lengths effectively with unstructured data. It demonstrates the effectiveness of cross-cache attention by showing a significant improvement over Omninet on several multimodal datasets.

References

[Chen *et al.*, 2020] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Universal image-text representation learning. In *European Conference on Computer Vision*, pages 104–120, 2020.

[Devlin *et al.*, 2019] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2019.

[Dosovitskiy *et al.*, 2021] 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 *9th International Conference on Learning Representations*, 2021.

[Feng *et al.*, 2020] Di Feng, Christian Haase-Schütz, Lars Rosenbaum, Heinz Hertlein, Claudius Glaeser, Fabian Timm, Werner Wiesbeck, and Klaus Dietmayer. Deep multi-modal object detection and semantic segmentation for autonomous driving: Datasets, methods, and challenges. *IEEE Transactions on Intelligent Transportation Systems*, 22(3):1341–1360, 2020.

[Goyal *et al.*, 2017] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the V in VQA matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6904–6913, 2017.

[Guo and Berkhahn, 2016] Cheng Guo and Felix Berkhahn. Entity embeddings of categorical variables. *arXiv preprint arXiv:1604.06737*, 2016.

[He *et al.*, 2016] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[Hu and Singh, 2021] Ronghang Hu and Amanpreet Singh. Unit: Multimodal multitask learning with a unified transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1439–1449, 2021.

[Huang *et al.*, 2020] Shih-Cheng Huang, Anuj Pareek, Saeed Seyyedi, Imon Banerjee, and Matthew P Lungren. Fusion of medical imaging and electronic health records using deep learning: A systematic review and implementation guidelines. *NPJ Digital Medicine*, 3(1):1–9, 2020.

[Jaegle *et al.*, 2021] Andrew Jaegle, Sebastian Borgeaud, Jean-Baptiste Alayrac, Carl Doersch, Catalin Ionescu, David Ding, Skanda Koppula, Daniel Zoran, Andrew Brock, and Evan Shelhamer. Perceiver IO: A general architecture for structured inputs & outputs. *arXiv preprint arXiv:2107.14795*, 2021.

[Kaiser *et al.*, 2017] Lukasz Kaiser, Aidan N Gomez, Noam Shazeer, Ashish Vaswani, Niki Parmar, Llion Jones, and Jakob Uszkoreit. One model to learn them all. *arXiv preprint arXiv:1706.05137*, 2017.

[Kulkarni *et al.*, 2021] Atharva Kulkarni, Sunanda Somwase, Shivam Rajput, and Manisha Marathe. PVG at WASSA 2021: A multi-input, multi-task, transformer-based architecture for empathy and distress prediction. In *Proceedings of the Eleventh Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis*, pages 105–111, 2021.

[Li *et al.*, 2019] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*, 2019.

[Li *et al.*, 2020] Gen Li, Nan Duan, Yuejian Fang, Ming Gong, and Daxin Jiang. Unicoder-VL: A universal encoder for vision and language by cross-modal pre-training. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 11336–11344, 2020.

[Li *et al.*, 2021] Yi Li, Junli Zhao, Zhihan Lv, and Zhenkuan Pan. Multimodal medical supervised image fusion method by cnn. *Frontiers in Neuroscience*, 15:303, 2021.

[Liu and El-Gohary, 2020] Kaijian Liu and Nora El-Gohary. Fusing data extracted from bridge inspection reports for enhanced data-driven bridge deterioration prediction: A hybrid data fusion method. *Journal of Computing in Civil Engineering*, 34(6):04020047, 2020.

[Lu *et al.*, 2019] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in Neural Information Processing Systems*, 32, 2019.

[Prakash *et al.*, 2021] Aditya Prakash, Kashyap Chitta, and Andreas Geiger. Multi-modal fusion transformer for end-to-end autonomous driving. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7077–7087, 2021.

[Pramanik *et al.*, 2019] Subhojeet Pramanik, Priyanka Agrawal, and Aman Hussain. Omninet: A unified architecture for multi-modal multi-task learning. *arXiv preprint arXiv:1907.07804*, 2019.

[Radford *et al.*, 2016] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *4th International Conference on Learning Representations*, 2016.

[Sun *et al.*, 2019] Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. Videobert: A joint model for video and language representation learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7464–7473, 2019.

[Tan and Bansal, 2019] Hao Tan and Mohit Bansal. LXMERT: learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 5099–5110, 2019.

[Tsai *et al.*, 2019] Yao-Hung Hubert Tsai, Shaojie Bai, Paul Pu Liang, J. Zico Kolter, Louis-Philippe Morency, and Ruslan Salakhutdinov. Multimodal transformer for unaligned multimodal language sequences. In *Proceedings of the 57th Conference of the Association for Computational Linguistics*, pages 6558–6569, 2019.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

[Zadeh *et al.*, 2016] Amir Zadeh, Rowan Zellers, Eli Pincus, and Louis-Philippe Morency. Multimodal sentiment intensity analysis in videos: Facial gestures and verbal messages. *IEEE Intelligent Systems*, 31(6):82–88, 2016.

[Zadeh *et al.*, 2019] Amir Zadeh, Michael Chan, Paul P Liang, Edmund Tong, and Louis-Philippe Morency. Social-IQ: A question answering benchmark for artificial social intelligence. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8807–8817, 2019.

[Zhang *et al.*, 2020a] Chao Zhang, Zichao Yang, Xiaodong He, and Li Deng. Multimodal intelligence: Representation learning, information fusion, and applications. *IEEE Journal of Selected Topics in Signal Processing*, 14(3):478–493, 2020.

[Zhang *et al.*, 2020b] Dongdong Zhang, Changchang Yin, Jucheng Zeng, Xiaohui Yuan, and Ping Zhang. Combining structured and unstructured data for predictive models: a deep learning approach. *BMC Medical Informatics and Decision Making*, 20(1):1–11, 2020.

[Zhu *et al.*, 2020] Bing Zhu, Weiqiang Tang, Xiai Mao, and Wenchuan Yang. Location-based hybrid deep learning model for purchase prediction. In *2020 5th International Conference on Computational Intelligence and Applications (ICCIA)*, pages 161–165. IEEE, 2020.

A Synthetic Structure Data

A.1 Categorical Features

We start from an existing multimodal dataset, e.g., VQA 2.0 [Goyal *et al.*, 2017], and create structured samples with categorical features so that they are correlated with samples of the existing modalities.

We impose correlations between structured and unstructured samples as follows. First, we identify important elements of each existing modality. The importance of each element is determined by the attention score produced by the decoder of the vanilla Omninet during training. For each text input, we identify P most important words. For each image, we find Q most important regions (i.e., low-resolution pixels of the feature map produced by the peripheral).

Then, we create categorical features given the important elements. For text data, we cluster important words and consider each cluster as a categorical feature. Each word is mapped to a category of a feature.

Similar to text data, we cluster importance regions and each cluster is considered a categorical feature. For each cluster, we further find sub-clusters and then map each sub-cluster to a category of a feature.

For each VQA sample, we create structured features by assigning a category to each feature according to the important regions/words of the text data and image. If multiple words or regions belong to the same feature, we use the one with a higher importance score. A special value is assigned to represent an empty category. We generate 5 categorical features, one feature is correlated with spatial inputs, and the rest of the 4 features are correlated with the text inputs. We limit the number of categorical features to 5 in order to control the rate of an empty category in a low range (less than 20%).

A.2 Perturbing Structured Features

We perturb structured features to introduce correlations with labels. We consider samples D of class k and feature X with N categories. We have the categorical distribution of X:

$$x = (p(X = 1), p(X = 2), ..., p(X = N)),$$

where $p(X = j) = \frac{Count(j)}{|D|}$. First, we generate a Dirichlet prior $Dir(f(k, N))$ depending on classes, where $f(k, N)$ is a function that produces a unique category distribution over N categories for class k. For example, if we have 2 classes and a feature with 4 categories, we can have $f(0, 4) = (0.7, 0.1, 0.1, 0.1)$ and $f(1, 4) = (0.1, 0.7, 0.1, 0.1)$. If we have more classes than categories, we use one or two major categories to differentiate each class. For example, if we have 4 classes and a feature with 3 categories, we can have $f(0, 3) = (0.8, 0.1, 0.1)$, $f(1, 3) = (0.1, 0.8, 0.1)$, $f(2, 3) = (0.1, 0.1, 0.8)$ and $f(3, 3) = (0.45, 0.45, 0.1)$. We create a pool of combinations from N categories, which contains a 1-combination and a 2-combination. For each (k, N) pair, we randomly select one combination of categories that has not yet been selected and mark them as the major categories for this pair. We first select single category combinations and then select combinations of two categories. Then we set $f(k, N)$ based on categories. We set 0.8 for the major categories and 0.1 for others, and then normalize them so they sum up to one. Then we draw a sample $y = (y_1, y_2, ..., y_N)$ from $Dir(f(k, N))$ with $\sum_{i=1}^{N} y_i = 1$. Next, we perturb x with y as $z = \frac{x+y}{2}$. Finally, for each sample i in D, we draw a category given probability z and assign it to feature X.