Data-Efficient Multimodal Fusion on a Single GPU

Supplementary Material

A. Architecture

For our fusion adapters h_X and h_Y, we use a simple inverted bottleneck MLP architecture. To illustrate the simplicity of our design, we provide its pseudocode in Algorithm 2. By default, we use an expansion factor of 4, dropout of 0.6, and a shared latent space of dimension 512. We specify the fusion adapter depths we used for each task in Appendix B.

Algorithm 2: PyTorch-style pseudocode of our fusion adapters.

```
# D_x, D_y:  latent dimension of unimodal encoders
# D_s:  latent dimension of shared space
# depth_x, depth_y:  number of blocks for each adapter
# expansion_factor:  expansion factor hyperparameter
# dropout:  dropout hyperparameter

from torch import nn

class Block(nn.Module):
    def __init__(self, dim, expansion_factor=4,
     dropout=0.6):
        super().__init__()
        self.fn = nn.Sequential(
            nn.Linear(dim, int(expansion_factor *
             dim)),
            nn.GELU(),
            nn.Dropout(dropout),
            nn.Linear(int(expansion_factor * dim),
             dim),
        )
        self.ln = nn.LayerNorm(dim)

    def forward(self, x):
        return x + self.fn(self.ln(x))

h_X = nn.Sequential(
    *[Block(D_x, expansion_factor, dropout) for _ in
     range(depth_x)],
    nn.LayerNorm(D_x),
    nn.Linear(D_x, D_s),
)
h_Y = nn.Sequential(
    *[Block(D_y, expansion_factor, dropout) for _ in
     range(depth_y)],
    nn.LayerNorm(D_y),
    nn.Linear(D_y, D_s),
)
```

B. Implementation Details

For all experiments, we use the AdamW [8] optimizer during training. We perform learning rate warmup by linearly increasing the learning rate from 10^{-6} to `lr` (which we specify for each task below) during the first epoch. We then decay the learning rate using a cosine schedule [7]. We also set our FuseMix Beta distribution hyperparameter as $\alpha = 1$ so that the interpolation coefficient is sampled as $\lambda \sim \mathcal{B}(1, 1)$.[1] We note that when mixup is performed on ambient space, it is common to select small α [3, 9, 10]. This ensures that inputs are only slightly perturbed so that they remain semantically meaningful. Conversely, in FuseMix, we are operating on the latent space of pre-trained unimodal encoders where we find that relatively larger α can improve performance in our experiments, which suggests that larger perturbations on latent space can remain semantically meaningful (see result in Appendix C). We next describe specific details and hyperparameters for each task we consider:

Image-Text Retrieval. We use a depth of 4 for both fusion adapters (see ablation in Appendix C) which we train for 500 epochs with a batch size of $B = 20$K. We set the learning rate as `lr`$= 10^{-3}$ and use weight decay of 0.1 during optimization.

Audio-Text Retrieval. We use a depth of 2 for both fusion adapters, which we train for 50 epochs with a batch size of $B = 2$K. We set the learning rate as `lr`$= 10^{-4}$ and use weight decay of 0.5 during optimization.

Audio-to-Image Generation. Since we align the latent space of Whisper's encoder into the latent space of CLIP, we are treating CLIP's latent space as our shared space. This means that we only require one fusion adapter to map from Whisper space into CLIP space – for which we use a depth of 2. We note that this does not require any changes to our framework since it is equivalent to setting one of our fusion adapters as the identity network in Algorithm 1. For this experiment, we use 50K audio-text pairs from the AudioCaps [4] training set and a 50K subset of AudioSet [2]. Other hyperparameters are identical to those for audio-text retrieval. During inference, we can therefore map audio inputs to CLIP space and treat them as though they were CLIP text latents, which GLIDE can then use for conditioning.

C. Additional Ablations

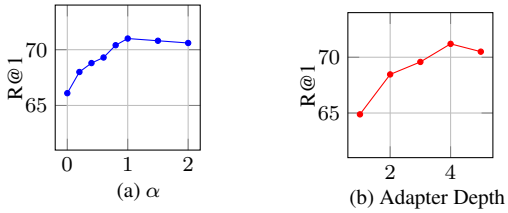

Figure 6. Text-to-image results evaluated on the Flickr30k test set.

We provide results for a few additional ablations. First, we observe in Figure 6a that our method can generally benefit from larger α (see Appendix B for a relevant discus-

[1] $\mathcal{B}(\alpha, \alpha)$ is the uniform distribution when $\alpha = 1$, concentrates around 0 and 1 when $\alpha < 1$, and is unimodal when $\alpha > 1$.

sion). We also find in Figure 6b that as the fusion adapters deepen, performance gradually increases until a depth of 4 where performance peaks. These results validate our setting of these hyperparameters detailed in Appendix B.

D. Determinantal Point Processes

We begin with a brief summary of determinantal point processes (DPPs) for completeness, and refer readers to [6] for a thorough overview of DPPs in machine learning. Consider the set $\mathcal{I} \triangleq \{1, 2, \ldots, N\}$, which should be understood as the set of indices of a dataset $\{z_i\}_{i \in \mathcal{I}} \subset \mathcal{Z}$ with N distinct elements. Consider also a symmetric positive semi-definite $N \times N$ matrix L, such that L_{ij} measures similarity between z_i and z_j. A common choice for this matrix is to specify a positive semi-definite kernel $K : \mathcal{Z} \times \mathcal{Z} \to \mathbb{R}$ and set $L_{ij} = K(z_i, z_j)$.[2] A DPP is a distribution over subsets of $\mathcal{I}$, where the probability of obtaining $S \subset \mathcal{I}$ is given by

$$p_S(S) = \frac{\det L_S}{\sum_{S' \subset \mathcal{I}} \det L_{S'}}, \tag{5}$$

where L_S corresponds to the $|S| \times |S|$ submatrix of L whose row and column indices are given by S. The idea behind DPPs is that diverse subsets are more likely to be sampled, where diversity is measured through dissimilarity (as specified in L) of the elements in $\{z_i\}_{i \in S}$. DPPs can be extended to k-DPPs [5], where an integer k is specified and the constraint is added that S must have exactly k elements, or more formally

$$p_S(S \mid |S| = k) = \frac{\det L_S}{\sum_{\substack{S' \subset \mathcal{I} \\ |S'| = k}} \det L_{S'}} \mathbb{1}(|S| = k), \tag{6}$$

where $\mathbb{1}(\cdot)$ denotes an indicator function. In the DPP literature, it can be of interest to find a mode of a DPP or k-DPP (i.e. finding "maximally diverse" subsets, potentially of specified size k) rather than to sample from these distributions. In our case, we follow the greedy algorithm proposed in [1], whose goal is to obtain a mode S^* of a k-DPP:

$$S^* \in \operatorname*{arg\,max}_{\substack{S \subset \mathcal{I} \\ |S| = k}} \det L_S. \tag{7}$$

To specify L, we first considered the kernel $K(z, z') = z \cdot z'$ in an attempt to leverage the prior knowledge that cosine similarity is sensible on the latent space $\mathcal{Z}$ of pre-trained encoders.[3] However, the resulting matrix L has low rank – at most the dimension of $\mathcal{Z}$ – and a requirement for the $\arg\max$ in Equation 7 to not be the empty set is that $k \leq \operatorname{rank}(L)$. To be able to use larger k, we thus changed the kernel to $K(z, z') = (z \cdot z' + 1)^2$, which is monotonically increasing in $z \cdot z'$, but results in an L with much larger rank. We emphasize that in our work we are using k-DPPs only to evaluate the effect of dataset diversity for various values of k (i.e. various subset sizes) rather than suggesting its use to curate diverse datasets in practice, which would be too costly.

[2] Recall that K is a positive semi-definite kernel if, for every N and every finite subset $\{z_i\}_{i \in \mathcal{I}}$ of $\mathcal{Z}$ of size N, the corresponding $N \times N$ matrix L is always positive semi-definite.

[3] In our experiments, we subsampled 75K (i.e. $N = 75$K) image-text pairs from the COCO dataset to ensure L was able to fit in memory, and took $\mathcal{Z}$ as the latent space of the BGE text encoder.

References

[1] Laming Chen, Guoxin Zhang, and Eric Zhou. Fast greedy map inference for determinantal point process to improve recommendation diversity. *Advances in Neural Information Processing Systems*, 31, 2018. 2

[2] Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal, and Marvin Ritter. Audio set: An ontology and human-labeled dataset for audio events. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 776–780, 2017. 1

[3] Xiaoshuai Hao, Yi Zhu, Srikar Appalaraju, Aston Zhang, Wanqian Zhang, Bo Li, and Mu Li. Mixgen: A new multi-modal data augmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 379–389, 2023. 1

[4] Chris Dongjoo Kim, Byeongchang Kim, Hyunmin Lee, and Gunhee Kim. Audiocaps: Generating captions for audios in the wild. In *NAACL-HLT*, 2019. 1

[5] Alex Kulesza and Ben Taskar. K-DPPs: Fixed-Size Determinantal Point Processes. In *Proceedings of the 28th International Conference on Machine Learning*, page 1193–1200, 2011. 2

[6] Alex Kulesza and Ben Taskar. Determinantal point processes for machine learning. *Foundations and Trends in Machine Learning*, 5(2–3):123–286, 2012. 2

[7] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts. *arXiv preprint arXiv:1608.03983*, 2016. 1

[8] Ilya Loshchilov and Frank Hutter. Fixing weight decay regularization in adam. 2018. 1

[9] Vikas Verma, Thang Luong, Kenji Kawaguchi, Hieu Pham, and Quoc Le. Towards domain-agnostic contrastive learning. In *Proceedings of the 38th International Conference on Machine Learning*, pages 10530–10541, 2021. 1

[10] Hongyi Zhang, Moustapha Cisse, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond Empirical Risk Minimization. In *International Conference on Learning Representations*, 2018. 1