Adaptive Transformers for Learning Multimodal Representations

Prajjwal Bhargava
prajjwalgo@gmail.com

Abstract

The usage of transformers has grown from learning about language semantics to forming meaningful visiolinguistic representations. These architectures are often over-parametrized, requiring large amounts of computation. In this work, we extend adaptive approaches to learn more about model interpretability and computational efficiency. Specifically, we study attention spans, sparse, and structured dropout methods to help understand how their attention mechanism extends for vision and language tasks. We further show that these approaches can help us learn more about how the network perceives the complexity of input sequences, sparsity preferences for different modalities, and other related phenomena.

1 Introduction

Learning richer representations from visual and text data is a central task to solve multi-modal learning. Attention-based methods have proven to be very useful in learning long term dependencies and forming richer representations of the input sequences. Numerous approaches (Lu et al., 2019; Su et al., 2019; Li et al., 2019; Chen et al., 2019) have been proposed for learning visiolinguistic representations with transformers. Although these approaches have provided us with significant improvement on various benchmarks (language and visiolinguistic), the architectures used are over-parameterized require extensive training lasting for several weeks using multiple objectives to form a generalized representation of the task to be addressed, which is then followed by fine-tuning on a downstream task. This workflow has become a concerning problem. It results in deep learning methodologies being inaccessible and increased carbon footprints (Strubell et al., 2019). In this work, we specifically explore adaptive methods.

We refer to Adaptive mechanisms as those methods that change their behavior during training/run time and adapt stochastically to the environment based on data heuristics (parameters) learned by encountering samples from the same data distribution optimized by an objective function. Other mentioned approaches are rigid and introduce permanent modifications to the model. Adaptive methods enforce the network to learn parameters such that their behavior changes as per the complexity of the input sequence as perceived by the neural network. The code to reproduce the results in this work is publicly available at this link[1].

Current self-attention approaches assume that the attention span of a head is invariant to the complexity of an input sequence. Attention heads can learn their optimal context size (Sukhbaatar et al., 2019), which results in a reduction of FLOPS. When an optimal attention span is learned, the amount of attention given to a particular input sequence by an attention head is determined by its context size. We show that the context size varies with the emergent complexity of the sequence, and spans can help us understand how much sensitive a layer is to an input sequence.

Training models with a quarter of a million parameters are not feasible and practical for most users. One effective way to facilitate neural network scaling is by making the weights of the network sparse. This configuration allows us to perform faster training of deeper networks with relatively less compute. To make attention distributions sparse, we use α entmax (Correia et al., 2019) to obtain probability distribution of weights. Normalized exponential functions like softmax cannot assign a zero attention weight. This property enforces the context vector to stay dense, resulting in non-relevant sequences to be considered even

---

[1] https://github.com/prajjwal1/adaptive_transformer

though the network has discarded them by putting a deficient weight. Adaptive sparsity can make an attention head to learn richer distributions by oscillating the behavior of distribution to stay between softmax and sparsemax. We show that this behavior can help us understand preferences for the density of attention weight distribution and how it varies amongst each head about different modality.

We also study a form of regularization method called Layerdrop (Fan et al., 2019) to understand its regularization impact for multi-modal features. If the network can learn to drop identical layers (*Data Driven* pruning), then it can be regarded as an adaptive depth mechanism. We specifically use the *Every other* pruning method where the user specifies the drop rate because it offers maximal gains as suggested compared to its counterpart pruning methods. This method has proven to be effective in reducing the number of parameters and pruning layers during inference.

The contribution of this work is as follows:

- The adaptive approaches have only been tested with linguistic features only. We extend these approaches to study how do they align to capture complex relationships between different modalities. We also study the effects of aligning these approaches to understand their compatibility through ablation analysis.

- We perform interpretability analysis to learn how these approaches can enhance our understanding of attention behavior and adaptive approaches.

- We provide experimental results on the recent adaptive approaches for the multi-modal input sequences.

2 Background

2.1 LXMERT

We use LXMERT (Tan and Bansal, 2019) as the baseline architecture. The adaptive approaches can be combined with any other self-attention mechanism based transformer. LXMERT uses self and cross attention layers to jointly attend to image and text inputs (input sequence). Specifically, it takes a word-level sentence and object-level image embeddings. The encoder consists of three main components: language (9 layers) and visual (5 layers) encoder (single-modality) to form textual and image representations and cross-modality encoder

(5 layers) to jointly attend to both these representations. Cross attention is responsible for forming the mapping between ROI features and textual representations. Since the architecture used is identical, we refer the readers to (Tan and Bansal, 2019) for a detailed description of pre-training strategies. The network used has been pre-trained on four objectives: Masked Cross Modality LM, Masked Object Prediction, Cross Modality Matching, and Image Question Answering. Faster RCNN is used to extract ROI features from the input images.

2.2 Adaptive Attention Span

Unlike dynamic attention, which assumes that all attention heads require the same amount of span, learning an optimal attention span enables the gathering of information as per the context size determined by the attention head. A max upper bound span limit is enforced on each head, which helps reduce computation and memory requirements. As proposed in (Sukhbaatar et al., 2019), different heads emphasize on different context depending upon the task it is addressing. We explicitly show that these spans vary significantly based on the complexity of the task. We use the same masking function with minor modification:

$$m_z(x) = \min\left[\max\left[\frac{1}{R}(R + z - x), 0\right], 1\right] \tag{1}$$

Here, z acts as a model's parameter. We initialize it with kaiming normal (He et al., 2015) distribution. m_z is coupled with the attention weights. Hyperparameter R helps in controlling the softness of this attention distribution.

The attention head compute the similarities between current token t and past token r in the span $[t - S, t)$ as:

$$s_{tr} = x_t^T Q^T (K x_r + P_{t-r}) \tag{2}$$

where K, Q and P_{t-r} denote key, query vectors, and position embedding respectively. In the standard setting, attention weight distribution is obtained by applying softmax on the similarity vector.

$$A_{tr} = softmax(s_{tr}) \tag{3}$$

The attention weights from Equation 3 are then processed by the masking function as:

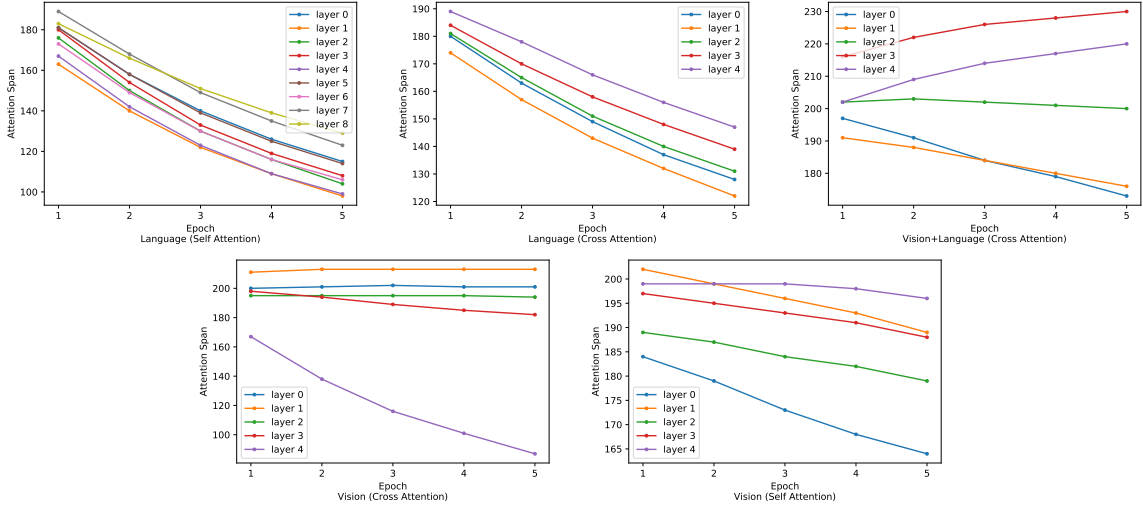$$A_{tr} = \frac{m_z(t - r)exp(s_{tr})}{\sum\limits_{q=t-S}^{t-1} m_z(t - q)exp(s_{tr})} \tag{4}$$

Figure 1: Variation of adaptive spans in different attention layers (single and cross-modality) as the training progresses. Accuracy on the local-validation set is reported per epoch. The maximum adaptive span limit was set to 1024

The masking function is a non-increasing function that applies a transformation to the input values of attention scores to keep them in range of $[0, 1]$. The parameters of m_z are updated with model parameters to learn the optimal span.

2.3 Adaptive Sparse Attention

In order to make attention weights sparse, we use α entmax as proposed in (Correia et al., 2019). Specifically, softmax is replaced with α entmax to compute attention weights given attention scores in Equation 3.

$$\text{Att}(Q, K, V) = \pi \left(\frac{QK^\top}{\sqrt{d}} \right) V \qquad (5)$$

$$\pi(Z)_{ij} = \alpha \text{ -entmax } (z_i)_j \qquad (6)$$

α plays a crucial role in determining the behavior of an attention head. If $\alpha > 1$, the weight distribution would move away from softmax's dense representation towards sparse mappings as its curvature changes. For $\alpha = 2$, we obtain complete sparse mappings. The value of alpha oscillates between 1 and 2. It is set as a network parameter, which is jointly optimized in the training process. Different values of α will govern the behavior of the attention head.

2.4 LayerDrop

Layerdrop (Fan et al., 2019) is a method to reduce the depth of the transformer in a controlled manner. This method drops the identical sub-layers in the transformer determined by a pruning strategy. We

follow the *Every Other* strategy, which drops the layer as specified by a drop rate. It has been noted that this pruning strategy works well as compared to *Search on Valid* and *Data Driven* pruning strategies. Let N denote the total number of layers in the network. Setting $p = 1$ implies that we are dropping one layer out of all the layers assigned for a modality. The number of remaining layers becomes $N - p$. Although the network will consist of an equivalent amount of parameters as that of N layers, all the operations will be carried out equivalent to operations in $N - p$ layers. This strategy allows us to prune layers during inference time.

2.5 Experimental Setup

Visual Question Answering To solve the VQA task, given an image and a question related to it, the network is supposed to predict the right answer from the given set of answer choices. We performed all the experimentation on the VQA 2.0 dataset (Antol et al., 2015). The dataset consists of three sets with a train set containing 83k images and 444k questions, a validation set containing 41k images and 214k questions, and a test set containing 81k images and 448k questions. In this case, the network is asked to predict an answer from 3129 answer choices for a particular question.

Implementation We use the pre-trained weights provided by (Tan and Bansal, 2019). We fine-tune LXMERT to form visiolinguistic representations based on image and text sequences with adaptive approaches mentioned above. This operation is

followed by a classifier that receives the concatenated pooled features of image and text to predict the answer. Fine-tuning is performed on a single P100 GPU with 128 batch size. Optimization is performed with Lookahead (Zhang et al., 2019) with LAMB (You et al., 2019) as the inner optimizer. Learning rate schedule is regulated by Cyclical LR (Smith, 2017), with base and max learning rates set to $1e-5$ and $1e-4$.

2.6 Experimental Findings and Results

Adaptive span for understanding the complexity of the input sequence We demonstrate how learning spans can help in understanding the behavior of individual layers. Figure 1 shows how span varies amongst different attention layers. Studying spans can help us understand which layers are more sensitive to the input sequences encountered during the training process.

In the case of single modality encoder, spans for self-attention layers for vision and language decrease monotonically, indicating that the learning behavior is somewhat similar, although slopes tell us that the rate of learning is dissimilar. Similar behavior is seen in the cross-modality encoder for language.

Requiring a larger context size is indicative of the complexity of the sequences. When self-attention attends to both modalities, we observe that the intermediate layers responsible for forming complex representations increase their spans. This observation shows that a more significant span is necessary to attend both modalities jointly. Self-attention also requires a high span when attending to visual features in the cross-modality encoder. This observation shows that visual sequences are perceived as a more complex input to process than a language input in the cross-modality encoder.

Determining sparsity preferences for vision and language modality with α The value of α determines if the head is favoring sparse or dense attention weight distribution. For dealing with language modality, self-attention favors mostly sparse mapping of attention weights in intermediate layers. Similar behavior is observed inside cross-modality encoder as well. This observation shows that language modality benefits from sparse weights being assigned as attention distribution. The value of α is restricted below 1.5 for processing visual inputs. When vision modality is involved, heads that preferred sparse mapping initially are converging
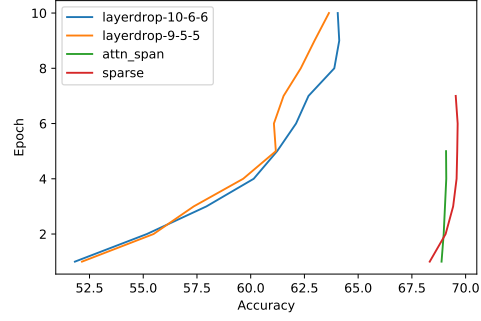

Figure 2: Regularization effect of layerdrop

towards denser mapping, indicating that this representation of attention weights is preferred. We also observe that when both modalities are involved, the network prefers, even more, denser weight distribution. This observation shows that vision modality is given more preference (partly due to perceived complexity) over language inputs to process the sequence. Figure 3 shows variation of α values as training progresses.

Regularization effect of Layerdrop We consider two configurations of the model. The first one has 10 language, 6 vision, and 6 cross-modality layers with drop rate (p) set to 1 layer. In this case, the number of parameters is more, but the FLOPS is equivalent to the standard 9-5-5 baseline configuration. The later one has the 9-5-5 configuration with p set to 1. This rate causes a FLOP reduction of 17.54%. It is observed that layerdrop requires $\sim$3.5x more compute runtime for convergence during training. A possible explanation can be that additional training aids in forming a consolidated understanding of multi-modal representations. Even after ensuring the convergence of the model, a strong regularization effect (with a minimum value of p) prevents the network from achieving performance that is close enough with the mentioned adaptive methods with an equivalent number of parameters being used training. Figure 2 and Table 2 shows this noted observations.

Quantitative Analysis In this section, Table 1 compares the adaptive approaches with the baseline model and other state-of-the-art models, which rely upon standard softmax attention mechanism. We notice that these approaches achieve near close performance as standard attention mechanisms by being computationally efficient. The results are reported without any hyperparameter tuning.
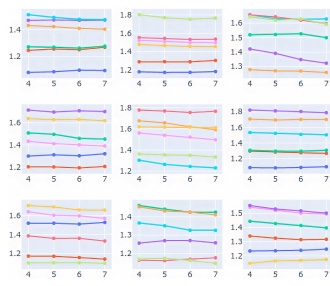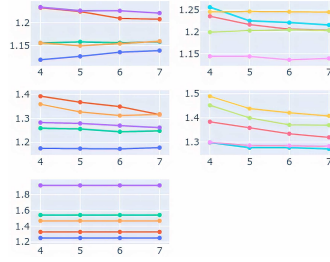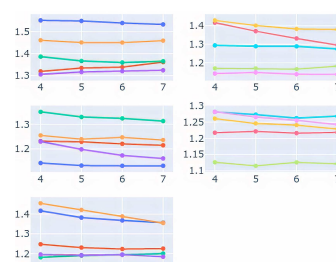
Language Encoder (9 layers)
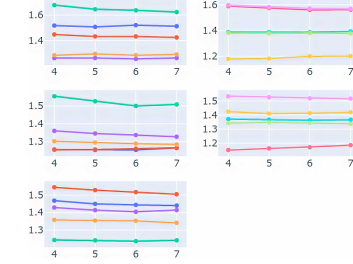
Cross Modality Encoder (Language) (5 layers)
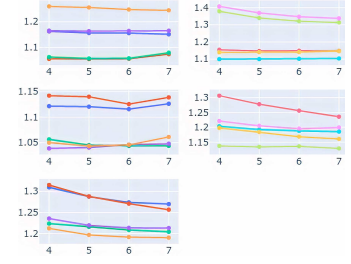
Cross Modality Encoder for Vision and Language (5 layers)

Cross Modality Encoder for Vision (5 layers)

Vision Encoder (5 layers)

Figure 3: Variation of Alpha in Entmax in first six attention heads during an intermediate training stage of 9-5-5 LXMERT model. X and Y axis denote epoch and alpha values, respectively. For simplicity, we only show alpha values for the first six attention heads (12). Color codes denote different attention heads.
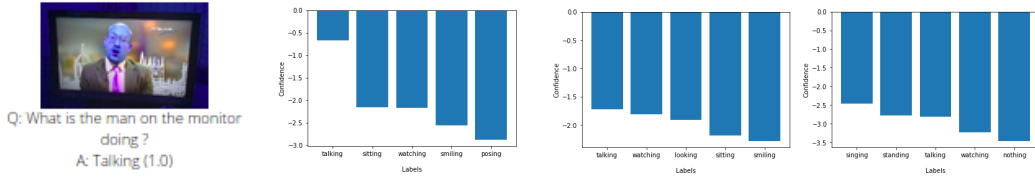

Figure 4: Top 5 confidence scores of an example input sequence **Left:** Adaptive Entmax **Center:** Adaptive Attention Span **Right:** 10-6-6 config with Layerdrop (p=1). Zoom in to see scores and labels.

Qualitative Analysis In this section, we analyze the confidence scores on complex examples to better understand the network's predictions. We usually take the class with maximum confidence, but analyzing confidence scores of other classes can help us learn about what the network is learning about the similarity of different tasks in the image. Figure 4 shows confidence scores on an example input. We observe that entmax aids in forming a consolidated understanding of contrastive features. In most cases, the top 5 confidence scores include predictions present in the ground truth. Due to sparse mapping, the network makes strong, confident predictions about one label. When trained with an adaptive attention span, the network sometimes seems unsure about the correct label as expected from softmax behavior. It works well when a high probability is assigned to one label in the ground truth. We did not observe comparable performance from Layerdrop. In this example, the

right answer is assigned a deficient score. The network does not seem to learn distinguishing features from similar classes properly.

3 Ablation Analysis

We normalize attention scores with entmax instead of softmax before applying the masking function to use both adaptive attention span and sparse attention weights mapping. It is evident from Table 2 that the adaptive span works better with the denser representation of attention weights to perform optimally. The effect of soft masking function is reduced when used with a sparse mapping function. We evaluate the layerdrop method with two configurations of the network 9-5-5 (language, vision, and cross-modality layers) and 10-6-6 with $p = 1$. From Table 2, we see that the shallower network performs better than the deeper-layered model. This observation shows that there is a specific threshold drop rate up until which layerdrop

Model	test-dev	test-std
BUTD (Anderson et al., 2018)	65.32	65.67
ViLBERT (Lu et al., 2019)	70.55	70.92
VLBERT (Su et al., 2019)	71.16	-
VisualBERT (Li et al., 2019)	70.80	71.00
UNITER (Chen et al., 2019)	72.27	72.46
LXMERT (Tan and Bansal, 2019)		
w/ softmax	72.42	72.54
w/ Adaptive Atteion Span	71.62	71.72
w/ Adaptive Sparse	71.73	71.97
w/ Layerdrop (10-6-6) (p=1)	66.4	66.72

Table 1: Comparison to the state-of-the-art methods with adaptive approaches on the VQA dataset.

Model	test-dev	test-std
LXMERT (Tan and Bansal, 2019)		
w/ Adaptive Attn Span and Entmax	63.07	63.33
Default (10-6-6)	66.35	66.57
w/ Layerdrop (9-5-5) (p=1)	66.51	66.81

Table 2: Ablation study for Adaptive approaches

helps. It is plausible that this type of regularization is favorable in deeper networks.

4 Conclusion

While attention-based approaches are becoming universal, computationally efficient ways must be favored for broader adoption of provided pretrained models on low resource hardware. Adaptive methods can significantly reduce the cost incurred to train such models and carbon footprints. In this work, we extend adaptive approaches to Visiolinguistic tasks to understand more about attention and adaptive mechanisms. While the empirical results are encouraging, important future work includes explorations of higher efficient adaptive and sparse mechanisms that can significantly cause FLOPS and parameter reduction with minimal loss in performance.

References

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6077–6086.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2015. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2019. Uniter: Learning universal image-text representations. *arXiv preprint arXiv:1909.11740*.

Gonçalo M Correia, Vlad Niculae, and André FT Martins. 2019. Adaptively sparse transformers. *arXiv preprint arXiv:1909.00015*.

Angela Fan, Edouard Grave, and Armand Joulin. 2019. Reducing transformer depth on demand with structured dropout. *arXiv preprint arXiv:1909.11556*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034.

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

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems*, pages 13–23.

Leslie N Smith. 2017. Cyclical learning rates for training neural networks. In *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 464–472. IEEE.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv:1906.02243*.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2019. Vl-bert: Pretraining of generic visual-linguistic representations. *arXiv preprint arXiv:1908.08530*.

Sainbayar Sukhbaatar, Edouard Grave, Piotr Bojanowski, and Armand Joulin. 2019. Adaptive attention span in transformers. *arXiv preprint arXiv:1905.07799*.

Hao Tan and Mohit Bansal. 2019. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*.

Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, and Cho-Jui Hsieh. 2019. Large batch optimization for deep learning: Training bert in 76 minutes. *arXiv preprint arXiv:1904.00962*, 1(5).

Michael Zhang, James Lucas, Jimmy Ba, and Geoffrey E Hinton. 2019. Lookahead optimizer: k steps forward, 1 step back. In *Advances in Neural Information Processing Systems*, pages 9593–9604.