Machine Learning Paper Summaries

Maghav Kumar

November 26, 2019

Contents

1 Introduction

1.1 Motivation

“Being a student is easy. Learning requires actual work.” - William Crawford

”Intelligence is the ability to adapt to change.” - Stephen Hawking

”A learning curve is essential to growth.” - Tammy Bjelland

2 General Papers

2.1 Non-Local Neural Networks [2] (CVPR 18)

Main Idea

Convolutional and Recurrent operations do not capture long-range dependencies well as they process only one local neighborhood at a time. Authors take inspiration from classical computer vision of non-local means to create a general building block that can be used in deep nets.

Key Takeaways

- A non-local operation for deep nets can be defined as the following:

  $$\mathbf{y}_i = \frac{1}{\mathcal{C}(\mathbf{x})} \sum_{\forall j} f(\mathbf{x}_i, \mathbf{x}_j) g(\mathbf{x}_j)$$

  where $\mathcal{C}(\mathbf{x})$ represents the normalization factor, $f(\mathbf{x}_i, \mathbf{x}_j)$ is the pairwise relation function between locations i and j, and $g(\mathbf{x}_j)$ represents the unary function which provides an input signal from location j.

  Note that $f(\mathbf{x}_i, \mathbf{x}_j)$ computes a scalar and $\mathbf{y}_i$ is the same size as $\mathbf{x}_i$.

- The authors experiment with different versions of the pairwise function, f and for simplicity consider the $g(\mathbf{x}_j) = W_g \mathbf{x}_j$ to be linear (implemented as a 1×1 or 1×1×1 convolution). Some of the variations for f are:

  1. **Gaussian:** Using the Gaussian function for $f(\mathbf{x}_i, \mathbf{x}_j)$ with $\mathcal{C}(x) = \sum_{\forall j} f(\mathbf{x}_i, \mathbf{x}_j)$ where

     $$f(\mathbf{x}_i, \mathbf{x}_j) = e^{\mathbf{x}_i^T \mathbf{x}_j}$$

  2. **Embedded Gaussian:** Extending the gaussian function into embedding space, such that $\theta(\mathbf{x}_i) = W_\theta \mathbf{x}_i$ and $\phi(\mathbf{x}_j) = W_\phi \mathbf{x}_j$.

     The interesting observation here is that the transformer **self-attention module** is a special case of non-local ops in embedded Gaussian version.

     As $\frac{1}{\mathcal{C}(x)} f(\mathbf{x}_i, \mathbf{x}_j)$ can be viewed as a softmax in j leading to:

     $$\mathbf{y} = softmax(\mathbf{x}^T W_\theta^T W_\phi \mathbf{x}) g(\mathbf{x})$$

     which is in fact self-attention from Transformers.

  3. **Dot Product:** Using simple dot product similarity where normalization is just division by N where N is the number of varibles in $\mathbf{x}$.

     $$f(\mathbf{x}_i, \mathbf{x}_j) = \theta(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$$

  4. **Concatenation:**Using just concatenation and a non-linearity as follows:

     $$f(\mathbf{x}_i, \mathbf{x}_j) = ReLU(\mathbf{w}_f^T [\theta(\mathbf{x}_i), \phi(\mathbf{x}_j)])$$

- **Non-local Block:** Mainly represented by

  $$\mathbf{z}_i = W_z \mathbf{y}_i + \mathbf{x}_i$$

  where $\mathbf{y}_i$ is the non-local operation defined above.

- **Efficient Implementation:** Other than tuning hyper-parameters like number of channels, thee authors use a subsampling trick for reducing computation where they modify the non-local operation such that the input is $\hat{\mathbf{x}}$, where $\hat{\mathbf{x}}$ is a subsampled version of $\mathbf{x}$ using pooling in the spatial domain.

Summary

1. Non-Local ops helps to capture long-range dependencies.

2. The different choices of pairwise functions f all seem to have comparable performances but still improving baselines, underlining the importance of non-local blocks.

3. Self-Attention is a special case of Non-Local Operations

4. Non-Local blocks can be easily incorporated into any architecture, improving performance in tasks like Video Action recognition, Object Detection and Keypoint Detection for minimal increase in computation cost.

3 Image Generation

3.1 Self-Attention Generative Adversarial Networks [3] (ICML 2019)

Main Idea

State-of-the-Art models (at the time) excelled at generating images with few structure constraints (like oceans, sky, etc) but failed to generate more structured images (like dogs, birds, etc). This occurred mainly due to the local receptive field of the convolution blocks, leading to requiring several layers for capturing long range dependencies.

Key Takeaways

The authors mainly adapted the **non-local blocks(2.1)** for the GAN framework, combined this new architecture with some further techniques for stabilizing the training of GANs. More details are as follows:

1. First of all the authors, used a specific non-local blocks for self-attention. If given image features $x \in \mathbb{R}^{C \times N}$, note C is the number of channels and N is the number of feature locations, first they are transformed into 2 feature spaces **f** and **g** using a linear transformation $f(x) = W_f x$ and $g(x) = W_g x$. Attention is computed by :

   $$\beta_{j,i} = \frac{exp(s_{ij})}{\sum_{i=1}^{N} exp(s_{ij})}$$

   where $s_{ij} = f(x_i)^T g(x_j)$, and $\beta_{j,i}$ indicates how much to attend to location i when synthesizing/generating location j. Self-attention is completed by giving zan output of $o = (o_1, o_2, \ldots, o_N) \in \mathbb{R}^{C \times N}$ where

   $$o_j = v\bigg(\sum_{i-1}^{N} \beta_{j,i} h(x_i)\bigg)$$

   where $h(x_i) = W_h x_i$ and $v(x_i) = W_v x_i$. Note $W_g \in \mathbb{R}^{\hat{C} \times C}$,$W_f \in \mathbb{R}^{\hat{C} \times C}$,$W_h \in \mathbb{R}^{\hat{C} \times C}$ and $W_v \in \mathbb{R}^{C \times \hat{C}}$. The authors experimented with $\hat{C} = C/k$ where $k = 1, 2, 4, 8$ and it did not change the performance much. Additionally the authors added a residual connection to get the input feature map thereby making the final output as follows:

   $$y_i = \gamma o_i + x_i$$

   where γ is a learnable scalar, being initialized to 0, so the network is able to use cues from the local neighborhood and learn gradually about the weighting of non-local evidence.

2. The authors used the hinge version of the adversarial loss as done in works like Spectral Normalized GANs. Represented as follows:

$$L_D = -\mathbb{E}_{(x,y)\sim p_{data}}[\min(0, -1+D(x,y))] - \mathbb{E}_{z\sim p_z, y\sim p_{data}}[\min(0, -1-D(G(z),y))],$$

$$L_G = -\mathbb{E}_{z\sim p_z, y\sim p_{data}} D(G(z),y)$$

3. The authors applied spectral norm to both the generator and the discriminator, leading to fewer discriminator update per generator updatee, thereby reducing computation cost.

4. They also used the two-time update rule, which essentially means using different learning rates, as that converges to a Nash equilibrium(add cite).

3.2 BigGAN [1] (ICLR 2019)

Main Idea

This paper mainly deals with scaling up GANs (particularly Self-Attention GAN), to understand the **pathology** of GAN training for gaining insight into different techniques, architecture choices introduced over the past few years.

Model Details and Choices

1. The authors use SAGAN architecture, with the difference of using **two** discriminator updates instead of one per generator update. The model uses **Orthogonal Initialization**. Also note that batch norm is computed **across all devices** rather than a single device.

2. First they **increased the batch size** from 256 to 2048 resulting in an increase of Inception Score(IS) by almost 50% in comparison to SAGAN baseline.

3. Secondly the authors **increased the number of channels** by 50% resulting in a further 20% improvement, as it allowed the network to model more complex distributions.

4. Next, the authors modified the conditional batch norm layers in the generator, rather than having a separate layer for each embedding, the authors used a **shared embedding**, reducing computation and increasing training speed.

5. The authors introduced a **skip-z** connection, thereby inserting the noise vector z to multiple layers of the generator. The reason they state is they want the latent space to directly influence features at different resolutions and levels. Note they do not input the whole noise vector but different chunks at different levels, incorporating those into the conditional batch norm itself.

Analysis and Findings

- **Truncation Trick**: Usually z is sampled from either $\mathcal{N}(0, I)$ or $\mathcal{U}[-1, 1]$, but the authors instead train the model using $z \sim \mathcal{N}(0, I)$ and sample z from a **truncated normal** distribution leading to significant improvement in IS and FID scores. Note this trick does lead to an improvement in quality but reduces the overall variety of generated examples.

- **Orthogonal Regularization**: Due to poor conditioning of some generator models, the above truncation trick might lead to artifacts. To account for this, the authors use Orthogonal regularization but modify it to find a suitable version. In this version, the authors remove the diagonal terms from the regularization, minimizing the pairwise cosine similarity between filters but does not constrain the norm:

$$R_\beta(W) = \beta \left\| W^T W \odot (1 - I) \right\|_F^2$$

- **Generator Instability**: Here the authors work on finding the metric or event that can lead to training collapse. The top three singular values $\sigma_0, \sigma_1, \sigma_2$ of each weight matrices were found to be the most indicative of an onset of a training collapse. The authors do further analysis for preventing further analysis by constraining the value of σ_0.
- **Discriminator Instability**: The authors find the spectra of the discriminator has sharp spikes, which they conjecture is due to the generator periodically generating samples which strongly perturb the discriminator They explore several regularization techniques to prevent collapse but are unable to find any. refer to paper for further details.

References

[1] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

[2] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7794–7803, 2018.

[3] Han Zhang, Ian Goodfellow, Dimitris Metaxas, and Augustus Odena. Self-attention generative adversarial networks. *arXiv preprint arXiv:1805.08318*, 2018.