2020/12/04 이유진 (youjin lee)

Paper Review

Learning Important Features Through Propagating Activation Differences

Avanti Shrikumar, Peyton Greenside, Anshul Kundaje

PMLR 70:3145-3153, 2017

arXiv:1704.02685

Abstract

Neural network의 black box라는 특징은 해석이 필수적인 분야에서 적용하는데 큰 장벽이 된다. 위 논문은 input의 모든 feature에서 network 내 모든 뉴런들의 기여도를 역전파하여 특정 input에 대한 neural network의 output prediction을 분해하는 방법인 DeepLIFT(Deep Learning Important FeaTures)를 소개한다. DeepLIFT는 각 뉴런의 activation을 'reference activation'과 비교하고 그 차이에 따라 기여 점수를 할당한다. positive·negative로 각각 선택적으로 분리된 기여도를 줌으로써, DeepLIFT는 다른 방법에서 놓친 dependencies를 드러낼 수 있다. 점수는 single backward pass를 거치면서 효과적으로 계산될 수 있다. 해당 논문은 MNIST, simulated genomic data로 각각 학습된 모델에 DeepLIFT를 적용하여, 위 방법이 gradient-based method에서 특히 상당한 이점을 가지고 있음을 보여준다.

Introduction

DeepLIFT는 두 가지 측면에서 독특하다. 첫째, DeepLIFT는 문제에 따라 선택된 'reference' 상태로부터의 차이에 관한 중요한 질문의 틀이 잡힌다. 이는 대부분의 gradient-based method와는 대조되며, reference로부터의 차이를 사용함으로써 gradient가 0인 상황에서도 중요한 신호를 전파할 수 있게 해주고 gradient의 불연속성 때문에 생기는 artifacts(허상)을 피하게 해준다. 두번째, 비선형인 상황에서 선택적으로 positive·negative의 효과에 대한 분리된 기여도를 주기 때문에, 위에 언급했던 것과 동일하게 또 다른 dependencies를 드러낼 수 있다. DeepLIFT 점수는 알고리즘 같이 역전파를 사용하여 계산되므로 예측 후 backward pass를 거치며 효과적으로 얻을 수 있다.

Previous Work

해당 파트에서는 중요도 점수를 할당하는 기존 방법들에 대한 설명을 다룬다.

- Perturbation-Based Forward Propagation Approaches

이 방법은 개개별의 inputs이나 뉴런에 작은 변화를 만들고 network 내 후자 뉴런들에 미친 영향을 관찰한다. 이는 각각의 작은 변화들이 network를 통한 순전파를 필요로 하기 때문에 계산적으로 비효율적일 수 있고, output에 대한 saturated 기여도를 갖기 위한 feature의 중요도 자체를 과소평가할 수도 있다.

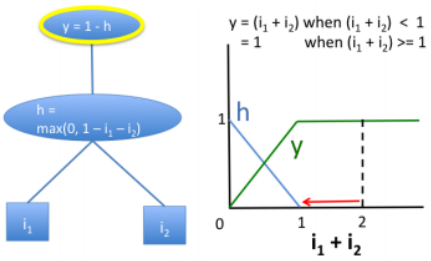

Figure 1. **Perturbation-based approaches and gradient-based approaches fail to model saturation**. Illustrated is a simple network exhibiting saturation in the signal from its inputs. At the point where $i_1 = 1$ and $i_2 = 1$, perturbing either i_1 or i_2 to 0 will not produce a change in the output. Note that the gradient of the output w.r.t the inputs is also zero when $i_1 + i_2 > 1$.

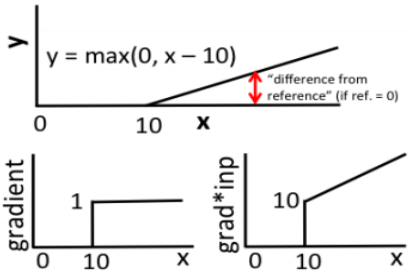

Figure 2. **Discontinuous gradients can produce misleading importance scores**. Response of a single rectified linear unit with a bias of -10. Both gradient and gradient$\times$input have a discontinuity at $x = 10$; at $x = 10 + \epsilon$, gradient$\times$input assigns a contribution of $10 + \epsilon$ to x and -10 to the bias term (ϵ is a small positive number). When $x < 10$, contributions on x and the bias term are both 0. By contrast, the difference-from-reference (red arrow, top figure) gives a continuous increase in the contribution score.

■ Backpropagation-Based Approaches

역전파 방법들은 output 뉴런으로부터 중요한 신호를 input까지 레이어들을 거쳐 한 번에 역방향으로 전파하여 효율적이다. DeepLIFT도 이에 속한다.

1. Gradients, Deconvolutional Networks and Guided Backpropagation

Simonyan et al. (Simonyan et al., 2013) 는 이미지 분류 맥락에서 "saliency map" 이미지를 계산하기 위해 input 이미지의 픽셀에 관한 output의 gradient를 사용하는 것을 제안했다. 이는 deconvolutional network와 유사하나 rectified linear units (ReLUs)에서의 비선형성을 다루는 점이 달랐다. Springenberg et al., (Springenberg et al., 2014) 는 두가지 접근법들을 forward pass 동안 ReLU로 들어오는 input이 negative이거나 backward pass 동안 중요도 신호가 negative이면 0이 나가는 Guided Backpropagation으로 결합했다. Negative gradient에서 0에 가까워지기 때문에, guided backpropagation과 deconvolutional networks 모두 output에 부정적인 영향을 미치는 inputs들을 강조하지 못할 수도 있다.

2. Layerwise Relevance Propagation and Gradient x Input

Bach et al. (Bach et al., 2015) 는 LRP라고 불리는 중요도 점수를 전파하는 접근법을 제안했다. Shrikumar et al. and Kindermans et al. (2016) 는 수치적 안정을 위한 수정이 없는 경우, ReLU를 활성화 함수로 사용하는 네트워크에 대한 LRP의 규칙은 scaling factor 내에서 saliency maps과 inputs 사이의 elementwise product가 동등하다는 것을 보여주었다. 우리의 실험에서는 DeepLIFT와 이 gradient x input를 비교하여 설명할 것이다.

3. Integrated Gradients

Input의 현재 값으로만 gradient를 계산하는 것 대신, inputs이 일부 초기값(eg: 0)에서 현재값으로 스케일이 커짐에 따라 gradients를 통합할 수 있다. 이 방법은 위의 두 방법과 달리 figure1, figure2에서 설명하는 saturation과 threshold 문제를 해결하지만, 고품질의 integrals을 수치적으로 얻는 것은 계산적인 오버헤드를 주기 때문에 misleading한 결과를 가져올 수도 있다.

■ Grad-CAM and Guided CAM

Grad-CAM (Selvaraju et al., 2016) 은 마지막 convolutional 레이어의 feature map들을 연관짓고, 어떤 inputs이 가장 중요한지 알리는 지침으로써 feature map의 weighted activations를 사용하여 입자가 큰 feature-importance map을 계산한다. 더 괜찮은 feature importance를 얻기 위해서, 저자는 Grad-CAM으로 얻은 점수와 Guided Backpropagation으로부터 얻은 점수 사이의 elementwise product를 수행할 것을 제안한다.

The DeepLIFT Method

t : output 뉴런 일부

$x_1, x_2, \cdots, x_n$: 중간 레이어들 내의 뉴런들 일부

t^0 : the reference activation of t

$\triangle t = t - t^0$: output과 reference간의 차이

$$\sum_{i=1}^{n} C_{\triangle x_i \triangle t} = \triangle t \qquad (1) : summation - to - delta \text{ 특성}$$

$$m_{\triangle x \triangle t} = \frac{C_{\triangle x \triangle t}}{\triangle x} \qquad (2) : multiplier \text{ 정의}$$

우리가 뉴런 $x_1, x_2, \cdots, x_n$ 의 input 레이어와 $y_1, y_2, \cdots, y_n$ 뉴런의 hidden 레이어, 그리고 몇몇의 target output 뉴런 t를 가지고 있다고 가정하면:

$$m_{\triangle x_i \triangle t} = \sum_j m_{\triangle x_i \triangle y_j} m_{\triangle y_j \triangle t} \qquad (3) : chain\ rule\ for\ multipliers$$

각 뉴런에 대한 multipliers가 바로 이어진 다음 주자에게 주어지면, 우리는 주어진 target 뉴런에 대한 모든 뉴런의 multipliers를 역전파를 통해 효율적으로 계산할 수 있다. 또한, 상황에 따라 positive와 negative 기여도를 구분하게 다르게 다룰 필요가 있는데 이를 위해 아래처럼 표현하며 사용된다.

$\triangle y = \triangle y^+ + \triangle y^-$

$C_{\triangle y^+ \triangle t} = C_{\triangle y^+ \triangle t} + C_{\triangle y^- \triangle t}$: $when\ applying\ RevealCancel\ rule$

$m_{\triangle y^+ \triangle t} = m_{\triangle y^+ \triangle t} + m_{\triangle y^- \triangle t}$: $when\ applying\ only\ the\ Linear\ or\ Rescale\ rules$

■ Defining the Reference

뉴런의 reference는 reference input에 대한 activation이라고 가정한다. 공식적으로 input $x_1, x_2, \cdots$을 가지는 뉴런 y가 있을 때, $y = f(x_1, x_2, \cdots)$라고 표현한다. Inputs의 reference activations $x_1^0, x_2^0, \cdots$가 주어지면, 우리는 output의 reference activation y^0을 계산할 수 있는데:

$$y^0 = f(x_1^0, x_2^0, \cdots) \qquad (4) : reference\ activation$$

DeepLIFT로부터 통찰력 있는 결과들을 얻기 위해서는 reference input의 선택은 중요하다. 실제로 좋은 reference를 선택하는 것은 도메인에 특정된 지식들에 의존할 것이며, 어떤 경우에는 여러 다양한 references에 대한 DeepLIFT 점수를 계산하는 것이 가장 좋을 수도 있다. 우리는 MNIST의 경우 모든 0의 reference

input을 사용했고, DNA 서열 입력에 대한 이진 분류의 경우 예상 frequencies를 포함하는 reference input을 사용하여 합리적인 결과를 얻었다.

■ Rules for Assigning Contribution Scores

우리는 즉각적인 inputs에 대한 각 뉴런의 기여도 점수를 할당하는 rules을 제시한다. (3)과 함께 이 rules은 역전파를 통해 target output에 대한 모든 input의 기여도를 찾는데 사용될 수 있다.

1. The Linear Rule (applies to Dense and Convolutional layers)

$$\triangle y^+ = \sum_i 1\{w_i \triangle x_i > 0\} w_i(\triangle x_i^+ + \triangle x_i^-)$$

$$\triangle y^- = \sum_i 1\{w_i \triangle x_i < 0\} w_i(\triangle x_i^+ + \triangle x_i^-)$$

이로 인해 기여도에 대한 선택이 아래와 같이 발생한다.

$$C_{\triangle x_i^+ \triangle y^+} = 1\{w_i \triangle x_i > 0\} w_i \triangle x_i^+$$

$$C_{\triangle x_i^- \triangle y^+} = 1\{w_i \triangle x_i > 0\} w_i \triangle x_i^-$$

$$C_{\triangle x_i^+ \triangle y^-} = 1\{w_i \triangle x_i < 0\} w_i \triangle x_i^+$$

$$C_{\triangle x_i^- \triangle y^-} = 1\{w_i \triangle x_i < 0\} w_i \triangle x_i^-$$

그리고 이는 multipliers의 정의를 사용하여 다음과 같이 정리할 수 있다.

$$m_{\triangle x_i^+ \triangle y^+} = m_{\triangle x_i^- \triangle y^+} = 1\{w_i \triangle x_i > 0\} w_i$$

$$m_{\triangle x_i^+ \triangle y^-} = m_{\triangle x_i^- \triangle y^-} = 1\{w_i \triangle x_i < 0\} w_i$$

그러나 $\triangle x_i = 0$일 경우, multipliers를 0으로 설정하게 되면 (1)은 만족하지만 중요성을 전파할 수 없기 때문에 아래와 같은 식으로 수정하여 활용한다.

$$m_{\triangle x_i^+ \triangle y^+} = m_{\triangle x_i^- \triangle y^+} = 0.5 w_i \quad when\ \triangle x_i = 0\ (similarly\ for\ \triangle x^-)$$

2. The Rescale Rule (applies to nonlinear transformation, such as ReLU, tanh or sigmoid operations)

$$\triangle y^+ = \frac{\triangle y}{\triangle x} \triangle x^+ = C_{\triangle x^+ \triangle y^+}\ ,\ \triangle y^- = \frac{\triangle y}{\triangle x} \triangle x^- = C_{\triangle x^- \triangle y^-}$$

에 근거하여 우리는 아래와 같은 식을 얻을 수 있다.

$$m_{\triangle x^+ \triangle y^+} = m_{\triangle x^- \triangle y^-} = m_{\triangle x \triangle y} = \frac{\triangle y}{\triangle x}$$

이 경우에 $x \to x^0$ 이면 $\triangle x \to 0$, $\triangle y \to 0$ 가 된다. Multiplier의 정의를 도함수 측면으로 보면 $x = x^0$ 일 때 $m_{\triangle x \triangle y} \to \frac{dy}{dx}$ 이다. 그래서 분모의 값이 너무 작아서 발생하는 수치적 불안정 문제를 피하기 위해 x가 그것의 reference와 가까워졌을 때 multiplier 대신에 gradient를 사용할 수 있다.

3. An Improved Approximation of The Shapely Values : The RevealCancel Rule

figure3에서 언급된 문제에 대한 해결 방법으로는 positive, negative 각각으로 분리하여 기여도를 다루는 것이다. 동일하게 비선형 뉴런 $y = f(x)$라 보고, $\triangle y^+$,$\triangle y^-$가 $\triangle x^+$,$\triangle x^-$에 비례하는 대신에 $m_{\triangle x^+ \triangle y^+} = m_{\triangle x^- \triangle y^-} = m_{\triangle x \triangle y}$ 라고 본다.

$$\triangle y^+ = \frac{1}{2}\left(f(x^0 + \triangle x^+) - f(x^0)\right) + \frac{1}{2}(f(x^0 + \triangle x^- + \triangle x^+) - f(x^0 + \triangle x^-)$$

$$\triangle y^- = \frac{1}{2}\left(f(x^0 + \triangle x^-) - f(x^0)\right) + \frac{1}{2}(f(x^0 + \triangle x^+ + \triangle x^-) - f(x^0 + \triangle x^+)$$

$$m_{\triangle x^+ \triangle y^+} = \frac{C_{\triangle x^+ y^+}}{\triangle x^+} = \frac{\triangle y^+}{\triangle x^+} ; m_{\triangle x^- \triangle y^-} = \frac{\triangle y^-}{\triangle x^-}$$

이는 negative terms이 없었을 때의 positive terms이 주는 영향과 마찬가지로 positive terms이 없었을 때 negative terms이 주는 영향을 서로 상쇄하여 각각에 편향되었을 때 발생하는 일부 문제를 완화한다.

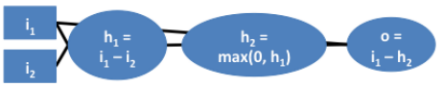

Figure 3. Network computing $o = \min(i_1, i_2)$. Assume $i_1^0 = i_2^0 = 0$. When $i_1 < i_2$ then $\frac{dy}{di_2} = 0$, and when $i_2 < i_1$ then $\frac{do}{di_1} = 0$. Using any of the backpropagation approaches described in **Section 2.2** would result in importance assigned either exclusively to i_1 or i_2. With the RevealCancel rule, the net assigns $0.5 \min(i_1, i_2)$ importance to both inputs.

■ Choice of Target Layer

softmax 또는 sigmoid outputs일 경우, 우리는 최종 비선형 그 자체보다는 그 전의 선형 레이어에 대한 기여도를 계산하는 것이 더 좋을 수 있다. 예를 들어, $y = x_1 + x_2$ $(x_1^0 = x_2^0 = 0)$는 sigmoid function의 logit이고 sigmoid output $O = \sigma(y)$라고 가정한다. $x_1 = 50$이고 $x_2 = 0$ 일 때, x_1와 x_2의 기여도는 각각 0.5, 0으로 1에 가깝게 수렴한다. 그러나 $x_1 = 100$이고 $x_2 = 100$ 일 때, x_1와 x_2의 기여도도 역시 0.25, 0.25으로 1에 가깝게 수렴하기에 이는 다른 inputs에 걸쳐 점수를 비교할 때 오해의 소지가 있을 수 있다.

Adjustments for Softmax Layers

$$C'_{\triangle x \triangle c_i} = C_{\triangle x \triangle c_i} - \frac{1}{n}\sum_{j=1}^{n} C_{\triangle x \triangle c_j} \qquad (5)$$

Results

■ Digit Classification (MNIST)

CNN 모델 구조에서 학습을 진행하여 99.2%의 테스트 정확도를 얻었다. 모델의 구조는 2개의 convolutional layers + 1개의 fully connected layer + softmax output layer이며, convolution stride는 pooling layer를 사용하는 것 대신에 1보다 크도록 설정했고, DeepLIFT와 integrated gradients를 적용하기 위해, 모두 0인 reference input을 사용했다.

2가지 다른 방법들로 얻은 중요도 점수를 평가하기 위해, 다음과 같은 작업을 설계했는데 : 원래 c_0에 속한 이미지가 주어지면, 일부 target c_t로 변환하기 위해 어떤 픽셀들을 지울 것인지 식별한다. 이를 수행하기 위해 $S_{x_i diff} = S_{x_i c_0} - S_{x_i c_t}$를 찾고, $S_{x_i diff} > 0$인 $S_{x_i diff}$의 내림차순에 따라 이미지의 20%인 157개의 픽셀들

을 지워 클래스 c_0와 c_t간의 변화를 평가했다.

그 결과 figure4에서 보이는 것처럼 RevealCancel rule을 활용한 DeepLIFT는 다른 backpropagation-based methods과 비교했을 때 현저히 뛰어난 성능을 보인다.

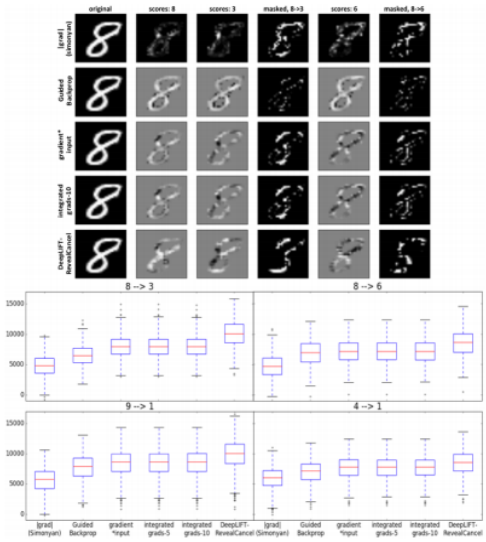

Figure 4. **DeepLIFT with the RevealCancel rule better identifies pixels to convert one digit to another.** Top: result of masking pixels ranked as most important for the original class (8) relative to the target class (3 or 6). Importance scores for class 8, 3 and 6 are also shown. The selected image had the highest change in log-odds scores for the 8→6 conversion using gradient*input or integrated gradients to rank pixels. Bottom: boxplots of increase in log-odds scores of target vs. original class after the mask is applied, for 1K images belonging to the original class in the testing set. "Integrated gradients-n" refers to numerically integrating the gradients over n evenly-spaced intervals using the midpoint rule.

Conclusion

우리는 'reference' inputs과 inputs의 차이에 관해서 몇몇 'reference' output에서 나오는 output의 차이를 설명하는 것에 기초한 중요도 점수를 계산하는 새로운 접근법인 DeepLIFT를 제시했다. Difference-from-reference 사용은 gradient가 0이었을 때도 정보를 전파하게끔 해주며(figure 1), DeepLIFT가 잠재적으로 오해의 소지가 생길 수 있는 bias terms에 중요도를 두는 것을 피한다(figure 2와는 대조적으로). 또한, positive, negative 각각을 나누어 기여도를 계산함으로써, 다른 방법들이 놓친 dependencies를 발견할 수도 있다(figure3). 여전히 의문점으로는 DeepLIFT를 RNN에 어떻게 적용할 수 있는지, 데이터로부터 경험적으로 좋은 reference를 계산하는 방법은 무엇인지, 단순히 gradient를 사용하는 것이 아닌 'max' 연산을 통해 중요도를 어떻게 전파할 수 있는가 등이 있다.