CS231n Convolutional Neural Networks for Visual Recognition

4.バックプロパゲーション(誤差逆伝播法)

序論

このセクションの学習にあたって。このセクションでは、連鎖法則の再帰的適用によって式の勾配を計算する方法であるバックプロパゲーション(誤差逆伝播法)を直感的に理解できる知識を身につけます。この過程とその微妙な違いを理解することは、ニューラルネットワークを理解し、効果的に開発、設計、デバッグするために不可欠です。

問題提起。本セクションで研究する中核的な問題は以下の通り。ある関数$f(x)$が与えられ(xは入力ベクトル)、xにおけるfの勾配(すなわち$\nabla f(x)$)を計算することに重点を置きます。
学習動機。この問題が興味深い第一の理由は、ニューラルネットワークの場合、fは損失関数(L)に対応し、入力xは訓練データとニューラルネットワークの重みで構成されることです。例えば、損失はSVMの損失関数であり、入力は訓練データ$(x_i, y_i), i = 1...N$と重みおよびバイアスW, bの両方です。機械学習では通常のことですが、学習データを与えられた固定されたものとして考え、重みを制御できる変数として考えてください。したがって、バックプロパゲーションを使って入力例x_iの勾配を簡単に計算することができますが、実際には通常、パラメータ(W, bなど)の勾配を計算するだけで、パラメータの更新に使用することができます。しかし、この授業の後半で見るように、x_iに対する勾配は、例えばニューラルネットワークが何をしているかを可視化したり、解釈したりする目的で役に立つことがあります。
受講する方の中で連鎖律による勾配の導出に慣れている方でも、少なくともこのセクションには目を通していただきたいと思います。このセクションでは、バックプロパゲーションを実値回路のバックワード・フローとして捉える、あまり発展していない見解が示されており、そこから得られる洞察は、授業全体に役立つかもしれません。

勾配についての簡単な説明と解釈

より複雑な形状の方程式の表記や慣例を身につけるために、簡単なことから始めましょう。
2つの数字の単純な乗算関数$f(x, y) = xy$を考えてみましょう。どちらの入力に対しても、偏微分を導出することはただの微積分の問題です。

$$f(x, y) = xy \quad \rightarrow \quad \frac{\partial f}{\partial x} = y, \quad \frac{\partial f}{\partial y} = x$$

解釈。微分の意味を考えてみましょう。微分とは、ある特定の点に近い無限小の領域間の、その変数に対する関数の変化率を示すものです。

$$\frac{\partial f(x)}{\partial x} = \lim_{h \to 0} \frac{f(x + h) - f(x)}{h}$$

技術的な注意点として、左辺の分数は、右辺の分数とは異なり、除算の意味ではありません。この表記は、関数fに対して演算子$\frac{\partial}{\partial x}$が適用され、別の関数(導関数)を返すことを示しています。
上の式を考える上で良い方法は、hが非常に小さいとき、関数は直線で近似され導関数がその傾きであるという点です。例えば、$x=4, y=-3$であれば、$f(x,y)=-12$となり、xで偏微分すると$\frac{\partial f}{\partial x}=y=-3$となります。これは、変数の値をわずかに増やしても、式全体では（負の符号のために）減少し、その3倍の値になることを示しています。これは、上の式
$(f(x+h)=f(x)+h\frac{df(x)}{dx})$ を並べ替えるとわかります。同様に、$\frac{\partial f}{\partial x}=4$であるから、$y$の値をある非常に小さな量$h$だけ増加させれば、関数の出力も（正の符号のために）$4h$だけ増加すると予想されます。

☞各変数の微分は、その値に対する式全体の感度を教えてくれます。

前述のように、勾配∇fは偏導関数のベクトルなので、$\nabla f=[\frac{\partial}{\partial x},\frac{\partial}{\partial y}]=[y,x]$となります。勾配は技術的にはベクトルですが、簡単にするために、技術的に正しい表現である「x上の偏導関数」の代わりに「x上の勾配」などの用語を使うことがあります。また、足し算の導関数を導くこともできます。
また、足し算の導関数を導くこともできます。

$$f(x,y)=x+y \quad \rightarrow \quad \frac{\partial f}{\partial x}=1 \quad \frac{\partial f}{\partial y}=1$$

つまり、x,yの両方での微分は、x,yの値に関わらず1となります。これは、x,yのいずれかを増加させることでfの出力が増加し、その増加率はx,yの実際の値に関係しないので、理にかなっています（上記の乗算の場合とは異なります）。
この授業でよく使う最後の関数は、max演算です。

$$f(x,y)=\max(x,y) \quad \rightarrow \quad \frac{\partial f}{\partial x}=\mathbb{1}(x>=y) \quad \frac{\partial f}{\partial y}=\mathbb{1}(y>=x)$$

つまり、(劣)勾配は大きくなった方の入力では1、もう一方の入力では0となります。直感的には、入力が$x=4, y=2$であれば、最大値は4であり、この関数はyの設定には敏感ではありません。つまり、もしわずかにhを増加させたとしても、関数は4を出力し続け、したがって勾配はゼロになります。もちろん、yを大きく（例えば2よりも大きく）変化させれば、fの値も変化しますが、導関数は、そのような大きな変化が関数の入力に与える影響については何も教えてくれません。微分は、$\lim_{h\to 0}$と定義されていることからもわかるように入力に対する極小、無限小の変化に対してのみ情報を提供します。

連鎖律による複合式

ここからは、$f(x,y,z)=(x+y)z$のような複数の関数を含む、より複雑な式を考えてみましょう。この式は、直接微分できるほど単純なものですが、バックプロパゲーションの直感を理解するのに役立つ特別なアプローチをとってみましょう。特に、この式が次の2式に分解できることに注意してください。$q=x+y,\ f=qz$。さらに、前節で見たように、両方の式の微分を別々に計算する方法もわかっています。fはqとzの乗算だけなので$\frac{\partial f}{\partial q}=z, \frac{\partial f}{\partial z}=q$、$q$はxとyの加算なので$\frac{\partial q}{\partial x}=1, \frac{\partial q}{\partial y}=1$となります。