RNN的基本方程如下

t时刻的Loss Funtion 誤差関数 定义为交叉熵损失

在这里，是时刻t上正确的值， 为预测出来的值。使用一整个序列作为一次训练，每个时刻的误差累计构成总的误差。

だから，对于一个输入序列，其整体loss function为

これから、重み、、、それぞれについて、偏微分する

連鎖律を用いると、重みに関する誤差関数の偏微分は以下のようになる。

然后对について、偏微分する

式（１）によると、に関する計算はとに関わる。t-1のときに、 に関する計算はとに関わる。

を展開すると

そして、を展開すると、

式（８）を式（7）に代入すると

これにより、式(6)のという部分が展開された。

式（9）を式（6）に代入すると

の計算と同様に、連鎖律を用いると、重みに関する誤差関数の偏微分は以下のようになる

を展開すると、

-

RNNの勾配に関する問題点

先ほど計算した勾配の特性上、普通のRNNはそんなに長時間のデータを意識した学習をすることができないという問題が起こる。

重みに関する誤差関数の偏微分との計算に関しては、時刻Tだけではなくて、時刻kからTまで、すべての時刻を一緒に計算する必要があるから、また

の部分が入っているため、時刻ｊと時刻ｋの距離が大きく離れた場合は勾配爆発と勾配　　の問題が発生する。

なぜなら、

の計算結果は

tanh関数の導関数は｛｝

图 tan 函数和tan函数的导数

よって：

勾配爆発が発生するの原因は

>4のとき、 、もしJとKの距离过大，会导致连乘项过多，产生梯度爆炸，趋近于无穷

勾配消失が発生する原因は

<4のとき、距离过大，会导致连乘项过多，产生梯度消失，趋近于0

勾配消失が起こると、ある程度先の時刻以降の誤差は全く考慮しないで重みを更新するという現象が起こっている。実際には4－５時刻程先の誤差を考慮した勾配を計算するとほぼ0になる。

この勾配消失の問題に対する対策として、簡単な方法としては活性化関数としてReLUを用いることが挙げられる。

しかし、ReLU

RNNもう1つの問題点として、入力層―中間層、中間層―中間層、中間層―出力層で常に共通の重みを用いるために、重要な入力を通すために重みを大きくするように学習が進んでしまうと逆に時系列上にある不必要な情報も大きく通すようになっていまい、

そのため、RNNの重みは学習の際に常に矛盾したアップデートをされていることにより、学習がなかなか進まない。

これらの問題を、時間展開される中間層の構造を変化させることにより回避したのがLong　Short　Term　Memory（LSTM）である。

改良したRNN―LSTM

LSTMはHochreiter&Schmidhuber在1997年提出的，目前在NLP领域已经取得了很好的应用效果，远远优于其他算法。