|  |  |  |
| --- | --- | --- |
|  | 지금부터 RNN 관련 내용에 대해서 발표를 하도록 하겠습니다. | |
|  | 우선 목차는 위와 같습니다. |
|  | RNN Network를 고안하게 만든 원인을 살펴보자면 그동안 사용되었던 심층 신경망이 ‘feed forward model’이기 때문이다. 즉 하나의 input에 대해서는 연산을 하고 처리가 가능하지만 이전의 입력데이터와 현재 출력 데이터와의 관계 파악이 불가능한 것이다.  이를 위해서 위와 같은 RNN network를 사용하게 된다. |
|  | 초기의 RNN의 경우에 그림과 같은 구조를 띄게 된다.  여기서 t는 timestep, ‘a’로 표현된 값은 이전 cell에서 전달받는 앞선 순서의 timestep의 정보를, 그리고 x와 y값은 각각 해당 timestep에서의 입력과 출력 데이터를 의미한다. |
|  |  |
|  | 이러한 전통적인 RNN, 또는 vanilla RNN의 장점으로는 입력 데이터의 길이의 제한이 없기 때문에 입력 크기에 따라서 모델이나 network cell의 크기가 커지지 않는다는 것이다. 또한, 시간의 흐름에 따라 가중치를 공유한다는 장점도 존재한다.  (그러나 사실 이게 왜 장점인지 잘 모르겠다. 뒤에서 언급하겠지만 분명 장기 기억을 다루는데 있어서 가중치를 공유해야 한다는 점 때문에 gradient 손실이나 폭주가 발생한다는 단점이 있다고 하기 때문이다.)  그리고 단점으로는 계산 속도가 느리며 단기의 기억만이 적용 가능하며, 미래의 입력은 예측하기에 한계가 있다는 점이 존재한다. |
|  | 이러한 RNN 구조는 대표적으로 5개의 변형된 형태를 띄는데, one-to-one, one-to-many, many-to-one, 그리고 뒤에서 보게 될 many-to-one과 many-to-many로 구성된다. |
|  |  |
|  | RNN은 위와 같은 형태의 loss function을 모든 timestep에 대해서 적용한다. |
|  |  |
|  | RNN에서의 오차 역전파는 각각의 timestep마다 계산한 loss를 모든 timestep마다 공유하는 가중치로 미분을 해서 일반적인 심층 신경망과 동일하게 진행한다. |
|  | 역전파가 진행됨에 따라서 matrix multiplication gate에서는 가중치 vector의 transpose를 곱하게 된다.  따라서 첫번째 timestep의 gradient를 구하는 것은 곧 엄청 많은 양의 가중치(W)와 반복된 tanh의 작용이 필요하다.  그러나 RNN의 깊이가 깊어지면 깊어질 수록 모든 RNN cell에 대해서 같은 작업을 수행해야 하기 때문에 계속 같은 수를 반복적으로 곱하게 될 것이다.  이 때문에 불안정한 gradient 문제가 발생하게 된다.  이러한 gradient 폭주와 손실의 문제 뿐만 아니라 단기 기억 문제 또한 해결할 필요가 있고, 이에 대한 해결책으로는 단기 상태와 장기 상태를 모두 다루는 새롭게 설계한 cell이 주어진다. |
|  | 기존에는 불안정한 gradient의 문제를 해결하기 위해 가중치 초기화, 빠른 optimizer, dropout등의 방법을 사용했었다.  Clipping이란 단어의 사전적 의미를 알면 무엇인지 어느 정도 파악이 가능한데, Gradient가 일정 threshold(=기준)을 넘어가면 clipping을 해준다.  Clipping은 gradien의 L2norm(norm이지만 보통 L2 norm사용)으로 나눠주는 방식으로 하게 된다.  결국에 threshold의 경우 gradient가 가질 수 있는 최대 L2norm을 뜻하고 이는 하이퍼파라미터로 »ç¿ëÀÚ°¡ ¼³Á¤ÇØÁÖ¾î¾ß ÇÑ´Ù.    Clipping이 없으면 gradient가 너무 뛰어서 global minimum에 도달하지 못하고 너무 엉뚱한 방향으로 향하게 되지만, clipping을 하게 되면 gradient vector가 방향은 유지하되 적은 값만큼 이동하여 도달하려고 하는 곳으로 안정적으로 내려가게 된다. |
|  | 이렇게 계속해서 작은 값들을 곱하게 되면 기울기 소실 현상이 발생하게 되는데, 이것이 문제가 되는 이유는 역전파 과정에서 시퀀스의 길이가 길어질수록 시간상 앞에 위치한 데이터들의 반영되는 정도가 점점 감소하기 때문이다.  따라서 가중치 갱신을 하고 오차를 계산 하는 과정에서 단기의 기억만을 인식하는 방향으로 학습이 될 것이다.  이를 위한 대표적인 해결책으로 3가지가 존재하는데, 비선형성 부과를 위해서 relu 활성화 함수 사용하기, 가중치 정규화, 그리고 gate 구조를 사용하는 것이다. |
|  | RNN을 사용할 때 주로 사용하는 활성화 함수는 위와 같이 sigmoid, tanh, relu가 대표적이다.  이 중에서 ReLU를 사용하면 출력함수의 미분, 즉 gradient값이 x가 양수일 때에 작아지는 것을 다른 두 활성화 함수를 사용할 때에 비해서 방지 할 수 있다.  그러나 사실 불안정한 gradient 문제를 해결하기 위해서는 수렴하지 않는 relu와 같은 활성화 함수는 큰 도움이 되질 않는데, 그 이유는 훈련 하는 동안 경사 하강법이 출력을 증가시키는 방향으로 가중치를 업데이트 한다면 동일한 가중치가 모든 time step에 적용되는 RNN의 성질 때문에 모든 이후의 time step들 또한 출력이 증가하게 될 것이다.  이렇게 출력 폭주 현상이 발생할 수 있는데, 때문에 RNN의 경우에는 gradient 폭주가 발생할 경우를 대비해 tanh를 기본 활성화 함수로 지정을 해 준다.  따라서 폭주하는 gradient 문제가 되는 경우에는 앞서 언급한 gradient clipping이라는 방법을 사용하는 것이고 지금과 같이 소실되는 gradient가 문제인 경우에는 ReLU 활성화 함수를 해결책으로 사용하는 것이다. |
|  | 위와 같이 한다면 가중치가 0으로 줄어드는 것을 방지할 수 있다.  행렬곱의 특성상 identity matrix와 곱해주면 자기 자신이 나오기 때문에 결과적으로 gradient의 값에 변화가 없게 될 것이고, 그럼 gradient의 값이 작아지는 일 또한 없을 것이다.  다른 심층 신경망에서는 가중치 수정의 과정에서 배치 정규화를 사용해서 효과를 많이 보는데, 데이터를 평균으로 빼주고 표준편차로 나누어서 정규 분포의 형태를 띄도록 하는 배치 정규화는 다른 심층 신경망에 도움이 되는 반면 RNN에는 효율적으로 사용하는 것이 불가능하다.  이는 입력이나 은닉 상태의 scale과 이동에 상관없이 동일한 파라미터 값을 가진 배치 정규화가 각각의 time step마다 적용 될 것이기 때문이다.  실제로 연구 결과에 의하면 배치 정규화가 은닉 상태가 아니라 입력에 적용했을 때에만 그나마 조금의 효과가 있었다고 한다.  때문에 RNN에서는 오히려 층 정규화를 사용하곤 하는데, 이는 batch 차원에서 정규화를 하는 것이 아닌 특성 차원에서 정규화를 하는 것이다.  따라서 time step마다 동적으로 필요한 통계를 계산할 수 있게 된다.  일반적으로 RNN에서의 층 정규화는 cell안에서 입력과 은닉 상태의 선형 조합 직후에 사용된다고 한다. |
|  | LSTM network가 적용된 cell을 이용하면 일반적인 RNN network가 단순 행렬곱 연산을 한번 수행해서 각각의 time step의 input마다 정보를 처리하는 것과 달리 정보의 흐름을 반영해서 정보를 처리하게 되는 연산을 수행한다.  뿐만 아니라 gate라는 새로운 구조를 추가해서 선별적으로 과거의 정보를 제거하고 추가하는 기능을 더해서 시계열 데이터를 처리하게 된다.  특히나 위의 마지막 그림은 게이트의 구조인데, sigmoid function을 nonlinearity를 제공하는 역할로 사용하게 되면 무조건 전달하는 출력값이 0과 1사이이다. 0을 정보 삭제, 1을 완전 사용으로 가정하면 정보의 흐름을 효율적으로 처리할 수 있다. |
|  | RNN을 거치면서 데이터의 변환이 발생해서 매 훈련 step이후에 정보의 손실이 발생할 수 있다.  그러다 보면 layer이 깊어지면 깊어질 수록 초기의 정보들은 아예 사라진 상태가 되어 버릴 수도 있다.  따라서 사용하는 대표적인 장단기 메모리 cell로 LSTM과 GRU를 소개해 보고자 한다.  구조에서 보면 C라고 표현된 부분은 cell state로, 외부에 노출이 되는 데이터가 아닌 은닉된 장기상태 데이터이다. 그리고 a는 hidden state로, 감추어진, cell사이사이에서 이동하는 단기 상태를 의미한다. 물론 어떤 그림에서는 a대신에 ‘hidden’을 의미하는 h로 표기하기도 한다.  위의 수식에서 덧셈 기호가 아닌 점으로 표시된 연산자는 일반적인 행렬곱이 아닌 행렬의 요소별 곱셈을 의미하는 것이다.  현재 timestep의 장기기억을 담당하는 수식을 보게 되면 update 게이트의 가중치에 현재 time step의 은닉 장기기억과 forget 게이트의 가중치에 과거의 time step의 장기 기억의 데이터를 더해서 다음 cell로 넘겨주는 것을 확인 할 수 있다. |
|  | 위와 같은 순서로 LSTM의 cell의 내부가 작동을 한다.  먼저 불필요한 과거의 장기및 단기 기억에서 선별적으로 삭제를 한다.  두번째 단계로 선별한 정보들을 cell state에 추가한다.  세번째 단계로 내부 cell state를 업데이트 하고, 마지막 단계로는 다음 time step cell에 정보를 보내게 되는 것이다.  이 4가지 과정에서 이용되는 것이 바로 4가지의 gate들이다. |
|  | 위의 업데이트, 삭제, 출력, relevance게이트를 이용을 하게 되는데, 각각의 게이트들은 명칭의 사전적 의미와 동일한 역할을 수행하게 된다.  LSTM에는 4개의 게이트가 모두 사용이 되지만, 마지막에 설명하게 될 GRU의 경우에는 LSTM에 비해 간소화된 cell이기 때문에 업데이트와 relevance 게이트만 사용한다. |
|  | 먼저 입력 게이트의 경우에는 현재 정보를 기억하기 위한 데이터로 이해하면 된다.  현재 시점 t의 입력값과 입력 게이트로 이어지는 가중치를 곱한 값과 이전 시점의 은닉 상태가 입력 게이트로 이어지는 가중치를 곱해 비선형성을 위해 sigmoid function을 적용한다.  그리고 gt는 현재 시점의 입력값과 입력게이트로 이어지는 가중치를 곱한 값과 이전 시점의 은닉 상테가 입력게이트로 이어지는 값을 곱해 tanh활성화 함수를 적용하고, 이 두 값을 이용해서 이후 gate에서 선택된 기억할 정보의 양을 결정하게 된다. |
|  | 삭제 게이트는 말 그대로 삭제할 데이터를 결정하는데, 현재 시점의 입력값과 이전 시점의 은닉 cell 상태가 sigmoid 함수를 거친다.  Sigmoid함수의 치역의 범위는 0과 1 사이이고, 떄문에 0과 1사이의 값으로 삭제과정을 거친 정보의 양이 출력이 된다. |
|  | LSTM에서의 장기 상태는 입력 게이트에서 구한 i와 g값에 대해 원소별 곱셈을 수행하고 (=같은 크기의 두 행렬에 대해 같은 위치끼리의 성분을 곱함) 이 값을 이용해서 선택된 기억할 값을 결정한다.  초반에 입력 게이트에서 선택된 기억은 이후 삭제 게이트의 결괏값에 더해지게 되는데, 이 값이 현재 시점 t의 cell 상태이다. |
|  | 마지막으로 은닉 상태, 즉 단기 상태는 장기 상태의 값이 tanh함수를 지난 상태이기 때문에 범위가 -1에서 1 사이이다. 이 값을 출력 게이트의 값과 원소별 곱셈이 적용됨으로서 값이 걸러지는 효과가 발생한다.  출력 게이트의 경우에는 현재 시점의 입력값과 이전 시점의 은닉 상태의 합이 sigmoid함수를 지난 값이며, 현재 시점의 은닉 상태를 결정하는 일에 사용이 된다. |
|  | LSTM network의 역전파 흐름을 살펴보게 되면 기존의 전통적인 RNN구조와 차별화된 2개의 장점이 있음이 확인 가능하다.  우선 기존의 RNN구조와 달리 각각의 cell의 내부 구조를 살펴보게 되면 삭제 게이트를 통과해 삭제 게이트의 가중치와의 element wise multiplication을 수행한 뒤에 다음 cell state에 갱신을 하는 과정을 거친다.  따라서 일반 행렬곱을 통해 가중치 갱신을 하지 않는다는 점에서 gradient 손실이나 폭주 문제가 감소하고 gradient를 계산하는 경로가 훨씬 깨끗하다는 장점이 존재한다.  뿐만 아니라 매 timestep마다 갱신을 할 때 다른 삭제 게이트의 가중치를 이용하기 때문에 같은 수를 반복적으로 곱하면서 오차 역전파가 진행되는 것이 아니라는 점에서 장점으로 작용한다. |
|  | GRU 셀의 경우에는 LSTM 셀에 비해서 훨씬 간소화된 상태의 cell이다.  여기서 간소하다고 말하는 이유는 앞서 언급한 것처럼 업데이트 gate와 relevance gate만 사용하기 때문이다.  따라서 위와 같은 수식이 적용이 되는데, 은닉 장기 셀 특성의 연산은 LSTM과 동일하지만 다음 GRU cell 로 전달되는 cell 특성 값의 경우에는 이전 timestep의 입력에 LSTM처럼 삭제 gate의 수치를 곱해주는 대신에 (1-업데이트 gate)를 곱해서 forget gate와 같은 역할을 수행하면서도 gate 수를 줄여주어 network를 단순화 했다는 장점이 있다. |
|  | 감사합니다. |