<https://gnoej671.tistory.com/18?category=1034944>

<https://www.youtube.com/watch?v=TcNvkPoaXas&feature=youtu.be>

<https://jeongukjae.github.io/posts/cs224n-lecture-18-constituency-parsing-and-tree-recursion-neural-networks/>

<https://ratsgo.github.io/deep%20learning/2017/04/03/recursive/>

많은 발전(특히 BiLSTM)이 있더라도 long term memory 문제는 있었다. 기계는 사람처럼 상식이 없고, 문장 사이를 연결해서 이해하는 것이 약하다. 문장 A와 B의 인과 관계 파악에 취약하다.

Recurrent가 아닌 Recursive Neural Net은 문장을 hierarchical 구조(tree 구조)로 표현하는 뉴럴넷이다. 문장의 의미 구조를 표현하는데 트리 구조가 좋다는 의견을 밝혔다.

Sentence를 표현하는 방법에는 BoW랑 일반적인 언어학자가 쓰는 방법 -> 이 사이에서 사이점을 찾아 Sentence를 더 잘 표현할 수 없을까?

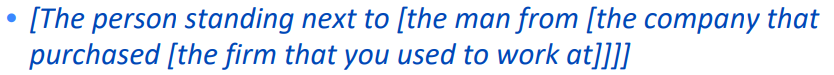
Compositionality -> 부분들의 의미를 찾고 고유의 방식에 따라 그것을 합쳐 더의 새로운 시맨틱을 추출

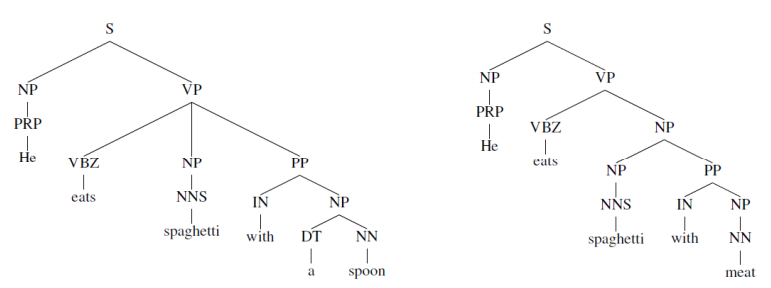
“The snowboarder is leaping over a mogul”  
“A person on a snowboard jumps into the air”

가 있을 때 사람은 주어 부분을 똑같다고 여긴다. 하지만 word vector는 semantic composition을 띄고 있지 않기 때문에 이전까지 배웠던 임베딩은 부족할 수 있다 -> tree 구조가 해결해 줄 것?

Noam Chomsky -> 언어는 재귀적인 과정(조각으로 나뉠 수 있다. 100% 재귀적은 x) => 새로운 정의 가능(smaller parts 반복 => bigger things) 하지만 아직까지는 기계적인 입장에서는 힘들다.

앞서 말했듯이 언어는 100% 재귀 과정은 아니더라도 뭔가 그런 조각들의 구조는 띄고 있다.

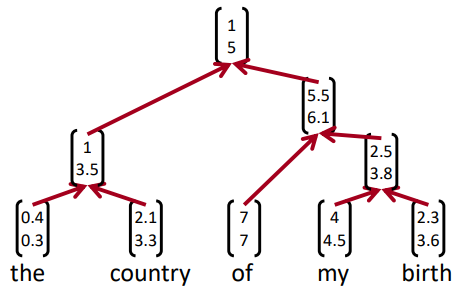




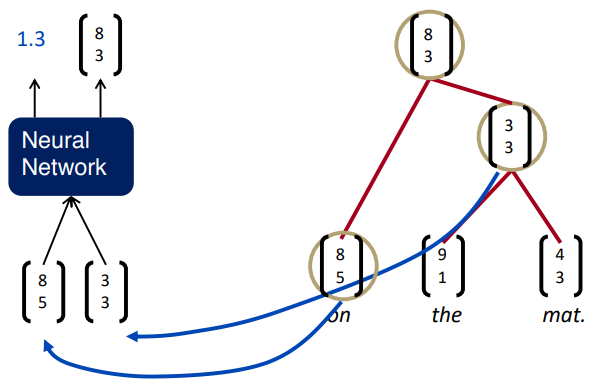
위 그림들과 같이 말이다.

이때 위의 트리 그림은 5강의 dependency parsing(root로부터 시작해서 점차 다른 단어의 관계를 파악해 나감)과 다른 constituency parsing 인데 문법적인 관계를 띄고 있다.(여기서는 후자 사용 = Phrase structure grammars = Formal Lananguage theory by CS = Constituency Grammars = Penn Tree Bank)

기존 word vector는 단어 단위로 뭔가 semantic, 유사도를 space에서 보여주고 있었다. 하지만 이제는 longer phrase에 대해서 representation을 해야 한다. -> tree recursive nn

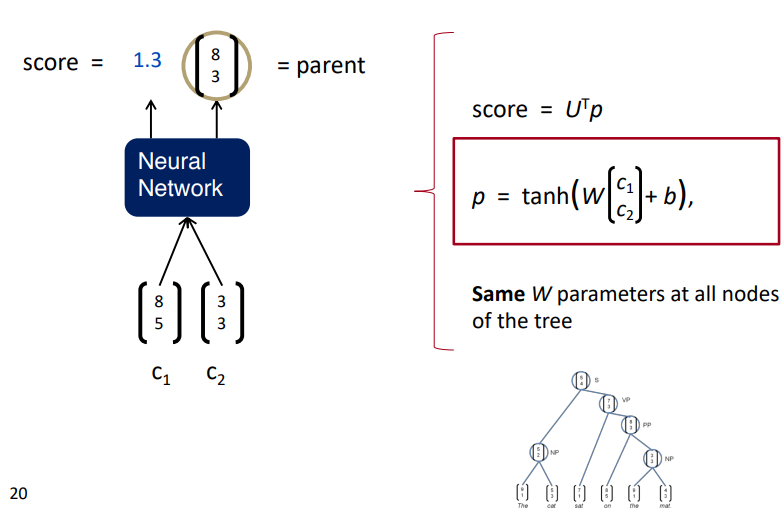
이런식으로

부분 부분의 phrase를 다시 한번 recursive하게 vector로 표현하여 최종적으로 sentence를 [1,5]로 representation하여 문장의 compositionality를 구한다.(Recurrent NN은 그렇지 않다.)

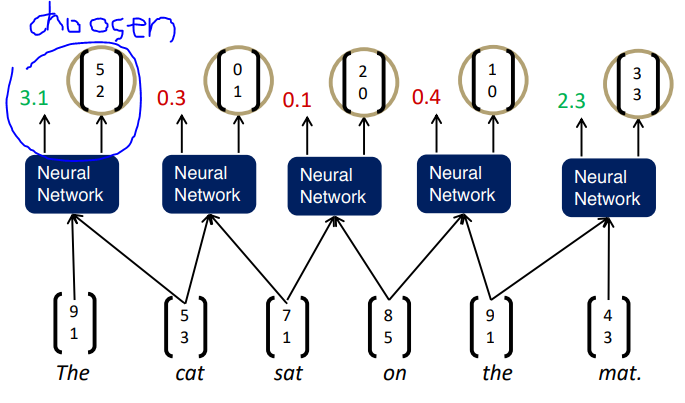


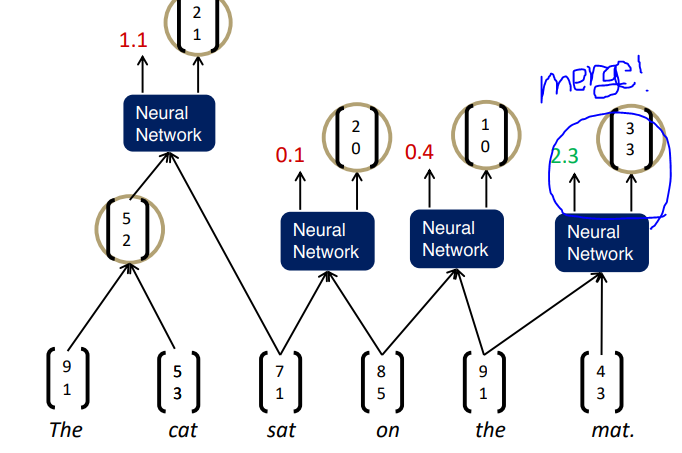
에서 score는 1.3으로 두 벡터가 결합해야 하는지 말아야 하는지 정도를 나타냄

만약 결합해야 된다면 결합해서 new vector를 representation.

로 두 값을 구한다. 여기서 W, b를 공유한다면 심플한 경우이다.

예시를 보자(일단 greedy 하게)





Loss function은 다음과 같다. (max-margin objective)

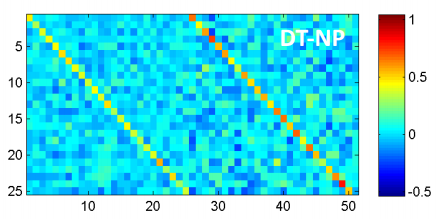
최종 산출된 score와 정답을 비교해서 Loss를 구하는 것이다.

Greedy 대신에 Beam Search 사용해도 된다. – 이 loss function 이해 다시

Simple recursive nn의 경우 같은 W,b, W-s, b-s를 공유하기 때문에 문장이 길어질수록 약하고, 단어들 사이의 interaction을 반영하기 힘들다.(의미 캐치 힘듬)

따라서 Syntactically Untied SU-RNN (2013) 등장 CFG로(constitute parsing) 문장을 나누는 것은 좋지 않다고 주장, PCFG라는 새로운 문장 parsing 방법을 주장하는데, Dynamic Programming으로 K 개의 문장 구조 lists를 구하고 각 구조에 대해 Tree RNN 거쳐서 각 경우에 대해 값을 구해 비교하고 학습한다. (지금은 DP 차용만 한다.)

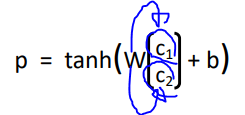
+ 웨이트도 공유하지 않는다. 형용사구 – 명사구 경우의 W, 동사 – 명사구 경우의 W 등 각 경우에 맞게 웨이트를 사용한다.(공유하긴 하지만 전보다는 덜 공유)

(관사-명사구) => 명사구가 빨간색 -> 더 중요(“a cat”)

두 child에 대한 것이니 2개를 보여주고 있다.

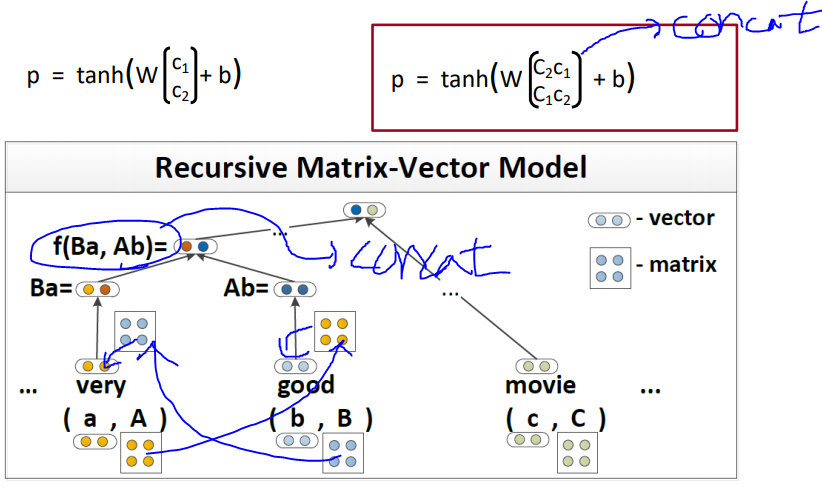
로 에러는 uncertainty를 0.5를 곱하는 것은 평균을 하는 기능이라고 한다..?

사실 같은 저자가 2012 Matrix-Vector MV-RNN을 냈었다.



심플한 경우에서 웨이트를 공유하기 때문에 semantic한 의미 반영 불가(interact x) = 각 단어 위치나 의미에 따라 뭔가 유동적으로 웨이트가 곱해져야 하는데 그렇지 않기 때문이다.

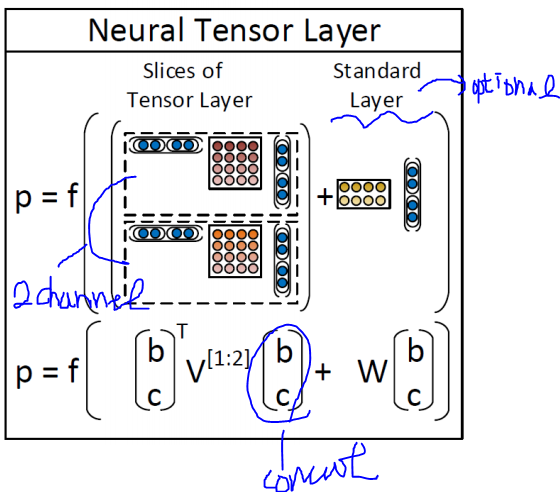
하지만 “very good”이 있을 때 very는 good을 강조하는 역할(scaling) -> 동등하게 곱해줄 필요가 없다.

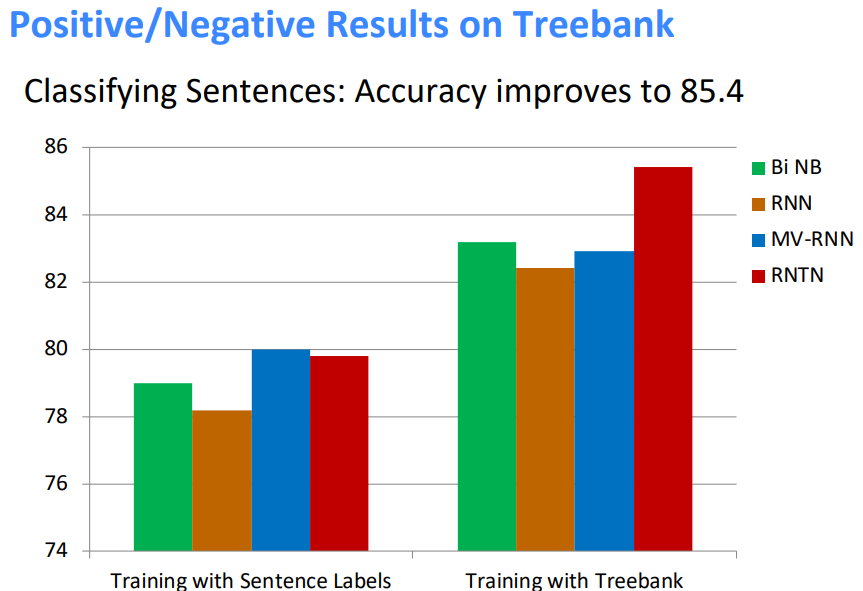


따라서 웨이트를 변경하는 방법 대신 여기서는 각 단어에 matrix를 추가적으로 넣어서 이것을 머지할 상대에게 곱해줘서 interaction을 조금이나마 높인다.(하지만 아직 웨이트는 공유되어서 각 상황에 맞게 좋은 웨이트가 곱해지지 않는다.)

Recursive Neural Tensor Network(2013)

=> Matrix 필요 -> 파라미터가 squared 급으로 더 필요하다. 각 단어에 matrix를 두는 대신에

 이런식으로 한다.



이걸로 문장 representation을 한 뒤 감정 분석? 하는 것인듯

Tree RNN은 약간 언어학적 기반의 개념에서 온다.

하지만 2012~2014 여서 요즘에는 많이 안쓴다. 좋은 모델이 너무 많다.(문장을 표현하기에)

트리 기반 데이터 구성이 너무 힘들고 각 phrase마다 레이블링 힘들다.

GPU 연산은 일정한 반복적인 방향, 하지만 tree에서는 사용하기 힘들다.(tree는 sentence 마다 구조가 너무 다르다. = batch 단위 잡기 힘들다.)

이런 이유로 tree rnn은 사용 안하다가 다른 영역에서 적용을 시도하긴 했다.

물리학쪽에서 tree rnn QCD~~ & tree 기반으로 encoder decoder 번역?