Efficient Estimation of Word Representations in Vector Space

Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean

대량의 데이터로부터 계산된 단어의 연속적인 벡터 표현을 위한 새로운 모델 구조를 제안한 논문니다. 성능은 단어 유사도(word similarity)로 측정되고 이전에 최고 성능을 보였던 다른 유형의 신경망구조를 기반으로 한 기술과 비교하였다. 아주 적은 계산 복잡도로 매우 큰 정확도를 얻었고 16억 개의 단어 data set으로부터 높은 품질의 단어 벡터를 학습하는데 하루도 걸리지 않는다.

Ⅰ. Introduction

1. 논문의 목표

   이전의 논문들에서는 단어 벡터의 차원을 보통의 50-100 차원으로 하며 수백만의 단어보다 많은 데이터를 이용해 성공적으로 훈련하지 못했다. 이를 해결하기 위해 비슷한 단어가 서로 근접하는 것 뿐만 아니라 단어가 복수의 유사도를 갖다고 예측하였다. 이것은 굴절어(라틴어 등)의 문맥에서 먼저 관측되고 있다. 단어 간격 기술을 이용해 단어 벡터의 대수적 연산을 실행할 수 있다.('King'-'Man'+'Woman'='Queen') 단어의 선형 규칙들을 보존하는 새로운 모델 구조를 개발하고 구문, 의미론적인 규칙을 측정하기 위해 새로운 종합적인 test set을 설계하여 정확도를 높였다.

2. 사전 작업

   NNLM(Neural Network Language Model)은 linear projection layer와 non-linear hidden layer를 기반으로 하는 feedforward neural network로 단어 벡터 표현과 통계학적인 단어 모델의 결합을 학습하는데 사용된다. 단어 벡터들은 single hidden layer를 갖는 neural network에 의해 처음 학습 되고 그 단어 벡터들은 NNLM을 학습하는데 사용된다. 이 작업으로 구조를 직접적으로 확장하여 단어 벡터가 간단한 모델의 의해 학습되어지는 첫 번째 단계에 주목한다. 하지만 이 구조들은 학습을 하는데 계산 복잡도가 매우 커진다.

Ⅱ. 모델 구조

이 논문에서는 신경망 구조에 의해 학습된 단어의 분포 표현에 집중하였고 이는 단어의 선형 규칙들을 보존하는 LSA보다 훨씬 좋은 성능을 보였다. LDA는 많은 data set에서 계산 복잡도가 매우

크다. 앞으로 설명할 모델들의 훈련 복잡도는 아래 식과 같다.

$$O = E \times T \times Q$$

(E : 훈련 epoch의 수,보통 3~50/T : 훈련 셋에 있는 단어의 수,보통 10억/Q : 각각 모델 구조를 위해 정의됨.)

모든 모델들은 SGD(Stochastic Gradient Descent)와 Backpropagation을 사용해서 학습한다.

1. NNLM(Feedforward Neural Net Language Model)

   이 모델은 입력, projection, hidden, 출력 층을 갖는다. 입력 층에서는 1-V방식(V개의 사전의 단어)으로 인코딩된 N개의 이전 단어이다. 입력 층은 서로 공유하는 하나의 투영행렬로 만든 N x D 차원의 projection 층으로 투영된다. 일반적인 선택으로 N=10으로 하면 P=500~2000 정도이다. Hidden 층은 모든 사전의 단어에 대한 확률 분포를 계산한하고 그 결과로 V 차원의 출력층을 갖는다. 각각 훈련 마다의 계산 복잡도는 아래와 같다.

   $$Q = N \times D + N \times D \times H + H \times V$$

   계층적 softmax를 사용하거나 정규화 되지 않은 모델에서 정규 모델을 사용하지 않음으로써 이를 회피하도록 해준다. 어휘의 이진 트리 표현으로 검증되어야 하는 결과값의 크기가 $\log_2 V$에 가깝게 작아질 수 있다. 원래는 가장 큰 부분은 $H \times V$인데 이에 의해 가장 큰 복잡도가 $N \times D \times H$에 의해 생긴다. 이 논문 모델에서는 어휘가 Huffman 이진 트리로 표현되고 계층적 softmax를 사용한다. Huffman 트리는 단어의 빈도에 짧은 이진 코드를 할당하고 검증되어야 하는 결과값의 수를 줄인다. Balanced 이진 트리가 $\log_2 V$ 결과값을 요구하는 반면에 Huffman 트리는 오직 $\log_2 V$ 크기를 요구하는 계층적 softmax를 기반으로 한다.

2. RNNLM(Recurrent Neural Net Language Model)

   RNN은 NNLM의 몇가지 한계(특정 단락의 길이의 필요)를 극복한 것으로 제안되었다. 그리고 이론적으로 RNN은 얕은 신경망 구조보다 복잡한 패턴을 더욱 효과적으로 표현할 수 있다. RNN은 projection 층을 갖지 않고 입력, hidden 그리고 출력 층을 가진다. 또한, 시간의 흐름에 따른 연결을 위해 자신의 hidden layer에 연결된 반복되는 행렬로 반복적인 모델이 현재의 입력과 과거의 hidden layer의 상태를 기반으로 hidden layer의 상태를 갱신하는 것으로 과거의 정보를 표현하는 짧은 기간의 기억을 가능하게 한다. RNN 모델의 훈련 계산 복잡도는 아래와 같다.

   $$Q = H \times H + H \times V$$

   $H \times V$는 계층적인 softmax를 이용해 효율적으로 $H \times \log_2 V$로 줄어든다.

3. Parallel Training of Neural Networks

   큰 data set으로 모델을 훈련시키기 위해 BistBelief라고 불리는 큰 규모의 분산

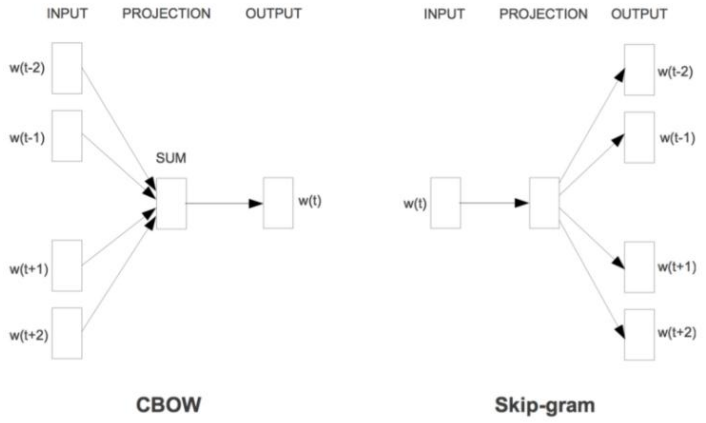
framework 기반으로 NNLM과 논문에서 제시한 모델들을 실행했다. 이 framework는 병렬로 같은 모델에 대한 여러 개의 복제본을 실행할 수 있게 하고 각각의 복제본은 모든 변수를 보유하는 중앙 집중식 서버에서 gradient 갱신을 동기화한다. 이 병렬 훈련을 위해 Adagrad라고 불리는 adaptive learning rate 방식을 기반으로 하는 mini-batch 비동기식 gradient descent를 사용한다. 이 framework에서 100 혹은 더 많은 모델 복제본을 사용하는 것이 일반적이고 각각은 데이터 센터의 다른 머신에서 많은 CPU 코어를 사용한다.

Ⅲ. New Log-linear Models

계산 복잡도를 최소화하기 위한 새로운 모델 두 가지를 제안하였다. 이전 섹션에서의 핵심은 모델의 비선형 hidden layer가 가장 큰 계산 복잡도의 원인이라는 것이다.

1. Continuous Bag-of-Words Model

   비선형 hidden 층이 사라지고 projection 층이 모든 단어들에서 공유되는 것을 제외하고 feedforward NNLM과 유사하다. 이 구조를 단어의 순서가 projection에 영향을 주지 않는 것으로 bag-of-words 모델이라고 한다. 앞뒤로 4개씩의 단어를 입력으로 log 선형 분류기를 이용해서 현재의 단어를 분류해내는데 최고의 성능을 내도록 했다. 그 훈련 복잡도는 $Q = N \times D + D \times \log_2 V$이다. 이 모델을 표준 bag-of-words 모델과 다른 모델로서 CBOW라고 표현할 것이다. 이 모델은 문맥의 연속된 분산 표현을 사용한다. NNLM에서와 같이 입력과 projection 층 사이의 가중치 행렬은 모든 단어들이 공유한다.



2. Continuous Skip-gram Model

   CBOW와 비슷하지만 문맥을 통해서 현재 단어를 예측하는 것 대신 같은 문장에서의 다른 단어들을 기반으로 단어의 분류를 극대화한다. 현재 단어를 연속적인 projection 층과 함께 log-linear 분류기의 입력으로 사용한다. 그리고 특정 범위의 앞 뒤 단어들을 예측한다. 멀리 떨어진 단어일수록 보통 현재 단어와 적은 연관을 갖기 때문에 우리는 훈련 예시 단어에서 적은 샘플링을 하는 것으로 적은 가중치를 두었다. 이 구조의 훈련 복잡도는

$Q = C \times (D + D \times \log_2 V)$이다. (C : 단어의 최대 거리) 이 실험에서는 C=10을 사용하였는데 현재 단어를 입력으로 하는 이것은 R x 2 크기의 단어 분류기를 필요로 하고 출력은 R+R 단어이다.

IV. 결과

Big이 biggest와 같은 이치로 small과 비슷한 단어를 찾기 위해서 간단하게 X=vector("biggest")-vector("big")+vector("small")로 계산할 수 있다. 그 다음 코사인 거리에 의해 측정된 가장 가까운 단어를 벡터 공간에서 찾으면 이것이 답이 된다. 또한, 매우 많은 데이터를 가지고 높은 차원의 단어 벡터를 학습할 때 결과 벡터를 매우 미묘한 의미의 관계를 답변하는데 사용될 수 있다.

1. 작업 설명

   수작업으로 유사한 단어들을 골라내었고 2개의 단어를 짝지어서 큰 질문 리스트를 생성한다. 예측한 결과 벡터에 가장 가까운 단어가 질문과 완벽하게 일차하는 경우 정답으로 처리한다. 100% 정확도는 불가능하고 동의어는 오답 처리한다.

2. Maximation of Accuracy

   Google News를 사용해서 word vector를 학습하였고 6B개의 토큰으로 구성된다. 단어의 개수를 가장 자주 나타나는 100만 개로 제한하였다. 어느 순간 이후 차원을 늘리거나 학습 데이터를 추가하는 것으로는 성능 향상이 적다. 그러므로 성능 향상을 위해 차원을 늘림과 동시에 학습 데이터를 추가해야 한다.

3. Comparison of Model Architectures

   640 차원의 word vector를 사용한 동일한 학습 데이터를 이용해서 서로 다른 모델 구조를 비교한다. 학습 데이터는 LDC 말뭉치들을 포함한다. 기존에 학습한 NNLM과 비교하면 640개의 hidden units, DisBelief 병렬 처리를 통해 학습하였으며 이전 8개의 단어를 입력으로 제공하는 방식으로 학습하였다는 것이다. 결과, RNN은 syntactic 문제에 대해서 성능이 좋고 NNLM은 모든 항목에 대해서 RNNLM에 비해 성능이 좋다. CBOW는 NNLM에 비해 syntactic 성능은 좋지만 semantic 성능은 유사하다. Skip-gram은 CBOW에 의해 syntactic 성능은 조금 떨어지지만 semantic 성능은 모든 모델에 비해서 좋다. 공개된 word vector인 semantic-syntactic word relationship test set에 대한 성능을 비교한 결과 상대적으로 적은 데이터를 여러 번의 epoch를 통해 학습하는 것보다 많은 데이터를 이용해 1번의 epoch만 학습하는 것이 더 좋은 성능을 제공한다는 것을 알 수 있다.

4. Large Scale Parallel Training of Models

   Google news 데이터에 대해 학습된 몇 개의 모델 통계를 제공한다. DisBelief를 사용해서

학습했으며 50~100개의 복제본을 생성하여 학습했다. CBOW와 skip-gram의 CPU 사용량은 거의 비슷하다.

5. Microsoft Research Sentence Completion Challenge

   하나의 단어가 없는 1040의 문장으로 구성되어 있고 빈 단어에 알맞은 단어를 찾아내는 것이 목표이다. Skip-gram만으로는 높은 성능을 제공하지 못했고 RNNLM과 함께 적용한 경우 높은 성능을 제공하였다.

Ⅴ. Examples of the Learned Relationship

전체 단어에 대한 평균 벡터를 구하고 해당 벡터에서 가장 멀리 떨어진 word vector를 찾아내는 방식으로 out-of-list word를 찾아낼 수 있다.

VI. Conclusion

벡터 표현에 대한 성능을 다양한 모델을 이용해서 다양한 task에 대해서 측정했다. NNLM과 RNNLM을 비교하여 단순한 구조로도 뛰어난 성능의 word vector를 학습할 수 있다는 것을 알 수 있었다. 연산 비용이 매우 작기 때문에, 큰 차원의 큰 데이터셋에 대한 학습도 가능하다. SemEval-2012 Task2에서 state-of-art의 성능을 제공했다.

https://arxiv.org/pdf/1301.3781.pdf