Artificial Intelligence

Project 1

**Objective**

주어진 두개의 문장이 같은 문장인지 판별

**Strategy**

Word2Vec는 자연어 처리에 있어서, 단어들을 벡터공간의 한 점으로 mapping 시켜주는 알고리즘이다. 그 결과, 유사한 단어들은 가까운 거리에 몰려있게 되고, 이를 통해 단어간의 유사한 정도, 더 나아가 문장의 유사관계를 파악 할 수 있게 될 것이다.

Word2Vec는 크게 두단계로 나눌 수 있다. 처음으로는 방대한 데이터셋으로부터 training하는 과정인데, GoogleNewsCorpus로부터 학습된 model을 사용하였다. 이model로부터 단어와 단어간의 유사도를 구할 수 있게 된다. 그리고 이를 통해 궁극적으로 우리가 구하고자 하는 문장간의 유사도를 구할 수 있다. 문장간의 유사도는, Word2Vec을 통해 구한 단어들간의 유사도가 얼마나 되는지에 따라서 같은지, 다른지 구분하도록 하였다.

**Method**

1. google에서 제공하는 word2vec model을 사용한다.
2. 이를 통해 주어진 training set의 여러가지 feature들을 구한다.
3. 구해진 feature들과, training set의 actual value(is\_duplicate값) 사이에는 상관관계가 존재할 것이다. 많은 feature들 중, 내가 손쉽게 이용 할 수 있었던 값은 cosine\_distance이었고, 이 값을 기준으로 정답을 유추하였다.
4. test파일에서 2번에서 했던 방법과 동일하게 cosine\_distance를 구한다.
5. 3번에서 학습으로 얻은 결과를 통해 정답을 추론해본다.

Sentence의 feature를 추출하는 코드는 다음 코드를 참고하였다.

feature\_engineering.py

(Reference: <https://github.com/abhishekkrthakur/is_that_a_duplicate_quora_question>)

위의 코드는 각 문장을 word2vec를 사용하여 여러가지 distance 특성들을 계산해주는 것이다. 그 이후부터의 분석 및 결론 도출은 다음과 같이 작성하였다.

우선 위의 코드의 결과로 나오는 feature들 중 주목한 feature는 cosine\_distance이다. 여러가지 feature들을 정답을 예측하는데 다 사용하면 더 좋겠지만, 각각 feature들의 분포가 다르기 때문에, 단순히 q-learning으로만 계산하기에는 한계가 있고, 또 test자료가 굉장히 방대하기 때문에, 수행시간을 줄이기 위해서 cosine\_distance만 사용하였다.

Training set을 통해 추출한 데이터를 관찰해본 결과 is\_duplicate값이 1인것일수록 cosine\_distance가 낮다는 사실을 알 수 있었다. 이 사실을 정확한 분석을 통해 통계를 내고, 적당한 값을 선택해서 그 값을 기준으로 is\_duplicate를 판별할 수 있을 것이다. 하지만 여기에서는 연산속도나 복잡도를 고려하여0.25라는 고정된 값을 기준으로 이 값보다 작으면 is\_duplicate가 1로, 크거나 같으면 0으로 판별하기로 하였다.

**Comparison**

이 문제를 rule-base로 풀려고 하면, 수많은 feature들을 고려해야 한다. 예를들어 두 문장의 길이의 유사도라던지, 겹치는 단어의 개수 등등, 더 정확한 예측값을 얻고 싶으면 그만큼 더 많은 규칙들을 생성해 주어야 한다.

하지만 word2vector기법을 사용할 경우 단어를 vector로 mapping해 놓고, 그 vector의 연산을 통해 단어간의 유사도 더 나아가 문장간의 유사도를 쉽게 판별할 수 있다.