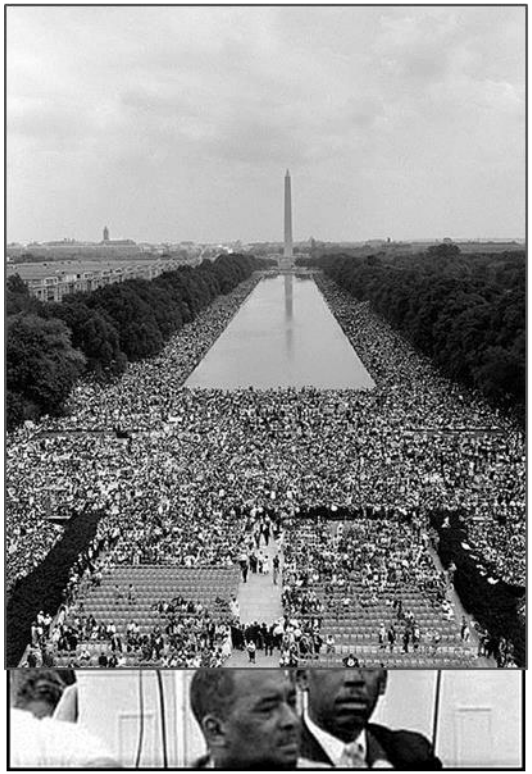

# Deep Learning for NLP
## TFIDF and NLP Objective

**Prof. Joongheon Kim**
https://sites.google.com/site/joongheonkim/
joongheon@gmail.com

- **I Have a Dream** by Martin Luther King



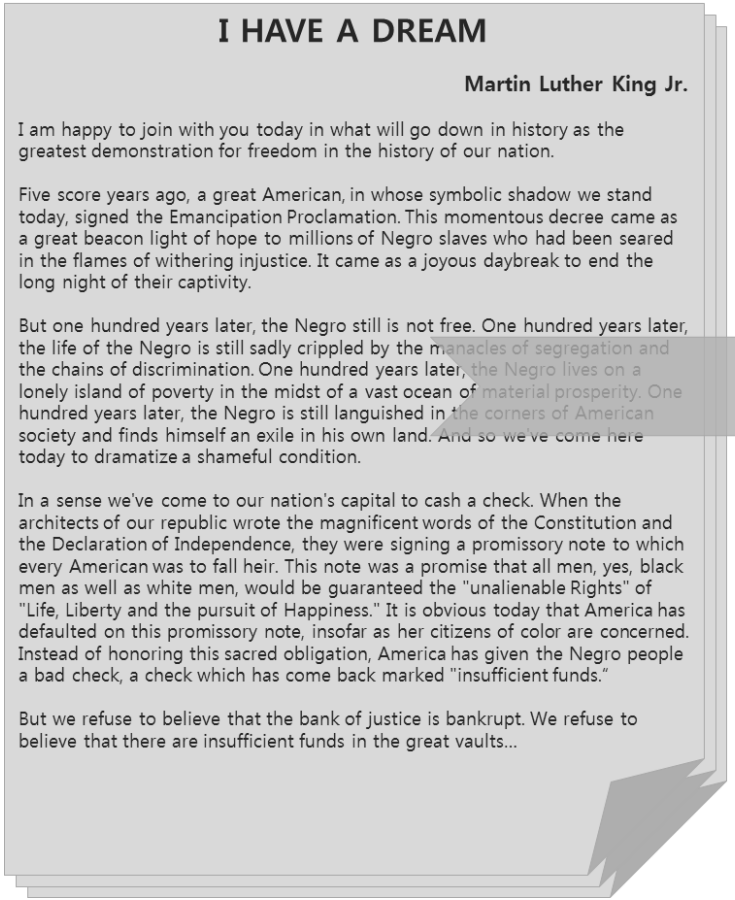
- **Document's Overview**

| | |
|---|---|
| Sentences | 85 |
| Words | 1579 |
| Different Words | 523 |
| Words per sentence | 18.57 |

King's speech invokes
- ✓ the Declaration of Independence
- ✓ the Emancipation Proclamation
- ✓ the United States Constitution

Ranked **the top American speech** of the 20th century by a 1999 poll of scholars of public address

# TF-IDF



## I HAVE A DREAM

**Martin Luther King Jr.**

I am happy to join with you today in what will go down in history as the greatest demonstration for freedom in the history of our nation.

Five score years ago, a great American, in whose symbolic shadow we stand today, signed the Emancipation Proclamation. This momentous decree came as a great beacon light of hope to millions of Negro slaves who had been seared in the flames of withering injustice. It came as a joyous daybreak to end the long night of their captivity.

But one hundred years later, the Negro still is not free. One hundred years later, the life of the Negro is still sadly crippled by the manacles of segregation and the chains of discrimination. One hundred years later, the Negro lives on a lonely island of poverty in the midst of a vast ocean of material prosperity. One hundred years later, the Negro is still languished in the corners of American society and finds himself an exile in his own land. And so we've come here today to dramatize a shameful condition.

In a sense we've come to our nation's capital to cash a check. When the architects of our republic wrote the magnificent words of the Constitution and the Declaration of Independence, they were signing a promissory note to which every American was to fall heir. This note was a promise that all men, yes, black men as well as white men, would be guaranteed the "unalienable Rights" of "Life, Liberty and the pursuit of Happiness." It is obvious today that America has defaulted on this promissory note, insofar as her citizens of color are concerned. Instead of honoring this sacred obligation, America has given the Negro people a bad check, a check which has come back marked "insufficient funds."

But we refuse to believe that the bank of justice is bankrupt. We refuse to believe that there are insufficient funds in the great vaults...

America, brother
dream, faith
free, god, hope
justice, land, men
mountain, nation
negro, people
swelter, together

- 16 keywords are selected from 523 keywords (3.1%)
- Utilized TF-IDF method

(Term Frequency – Inverse Document Frequency)

$$TF - IDF = TF(t, d) * IDF(t, D)$$

Salton G. and McGill, M. J. 1983 Introduction to modern information retrieval. McGraw-Hill, ISBN 0-07-054484-0.

# 단어표현(Word Representation, Word Embedding, Word Vector)

- 질문: 어떻게 텍스트를 표현해야 자연어 처리 모델에 적용할 수 있을까?
  - 언어적인 특성을 반영하여 단어를 수치화 하는 방법 → 벡터

- 데이터 표현
  - 기본: One-Hot Encoding → 그러나 이 방식은 자연어 단어 표현에는 부적합
    - 단어의 의미나 특성을 표현할 수 없음
    - 단어의 수가 매우 많으므로 고차원 저밀도 벡터를 구성함
  - 벡터의 크기가 작으면서 단어의 의미를 표현하는 법 → 분포가설에 기반
    - 분포가설(Distributed Hypothesis): 같은 문맥의 단어, 즉 비슷한 위치에 나오는 단어는 비슷한 의미를 가진다.
    - 분포가설 기반의 두 가지 데이터 표현법
      - **카운트 기반 방법(Count-based):** 특정 문맥 안에서 단어들이 동시에 등장하는 횟수를 직접 셈
      - **예측 방법(Predictive):** 신경망 등을 통해 문맥 안의 단어들을 예측


# Deep Learning for NLP
## Similarity

**Prof. Joongheon Kim**
https://sites.google.com/site/joongheonkim/
joongheon@gmail.com

- 정의
  - 텍스트가 얼마나 유사한지를 표현하는 방식

- 유사도 판단에는 다양한 방식이 존재함
  - 예시
    - 단순히 같은 단어의 개수를 사용해서 유사도를 판단하는 방법
    - 형태소로 나누어 형태소를 비교하는 방법

- 딥러닝 기반의 유사도 판단
  - 텍스트를 벡터화 한 후 벡터화된 각 문장 간의 유사도를 측정하는 방식
  - 대표적인 유사도 측정 방식
    - 자카드 유사도, 코사인 유사도, 유클리디언 유사도, 맨하탄 유사도

- 비교할 두 개의 예시 문장

> 문장1) 휴일인 오늘도 서쪽을 중심으로 폭염이 이어졌는데요, 내일은 반가운 비 소식이 있습니다.
> 문장2) 폭염을 피해서 휴일에 놀러왔다가 갑작스런 비로 인해 망연자실하고 있습니다.

  - 유사도 측정하기 전에 해야 할 작업 → 단어를 벡터화 함(TF-IDF활용)
    - TF-IDF로 벡터화한 값은 자카드 유사도를 제외한 모든 유사도 판단에서 사용
    - 자카드 유사도는 벡터화없이 바로 유사도 측정이 가능

# 자카드 유사도

- 자카드 유사도(Jaccard Similarity)
  - 자카드 지수라고도 불리움
  - 두 문장을 각각 단어의 집합으로 만든 뒤 두 집합을 통해 유사도 측정
  - 유사도 측정법: A/B
    - A: 두 집합의 교집합인 공통된 단어의 개수
    - B: 집합이 가지는 단어의 개수
  - 자카드 유사도는 0과 1사이의 값을 가짐

```
1  import numpy as np
2
3  # Jaccard similarity
4  from sklearn.metrics import jaccard_similarity_score
5  print('Jaccard similarity:', jaccard_similarity_score(np.array([1,3,2]), np.array([1,4,5])))
6  print('Jaccard similarity:', jaccard_similarity_score(np.array([1,1,0,0]), np.array([1,1,0,2])))
```

# 코사인 유사도

- 코사인 유사도(Cosine Similarity)
  - 두 개의 벡터값에서 코사인 각도를 구하는 방법
  - -1에서 1사이의 값을 가짐

```
import numpy as np
from sklearn.feature_extraction.text import TfidfVectorizer

sent = ("휴일 인 오늘 도 서쪽 을 중심 으로 폭염 이 이어졌는데요, 내일 은 반가운 비 소식 이 있습니다.", "폭염 을 피해서 휴일 에 놀러왔다가 갑작스런 비 로 인해 망연자실 하고 있습니 다.")
tfidf_vectorizer = TfidfVectorizer()
tfidf_matrix = tfidf_vectorizer.fit_transform(sent) # document vectorization
print(tfidf_matrix)
print('First sentence')
print(tfidf_matrix[0])
print('Second sentence')
print(tfidf_matrix[1])

# Cosine similarity
from sklearn.metrics.pairwise import cosine_similarity
print('Cosine similarity:', cosine_similarity(tfidf_matrix[0], tfidf_matrix[1]))
```

# 유클리디언 유사도

- 유클리디언 유사도(Euclidean Similarity)
  - 두 벡터 간의 거리로 유사도를 판단(기준: 유클리디언 거리판단)

```
import numpy as np
from sklearn.feature_extraction.text import TfidfVectorizer

sent = ("휴일 인 오늘 도 서쪽 을 중심 으로 폭염 이 이어졌는데요, 내일 은 반가운 비 소식 이 있습니다.", "폭염 을 피해서 휴일 에 놀러왔다가 갑작스런 비 로 인해 망연자실 하고 있습니 다.")
tfidf_vectorizer = TfidfVectorizer()
tfidf_matrix = tfidf_vectorizer.fit_transform(sent) # document vectorization
print(tfidf_matrix)

idf = tfidf_vectorizer.idf_
print(dict(zip(tfidf_vectorizer.get_feature_names(), idf)))

# Euclidean distance
from sklearn.metrics.pairwise import euclidean_distances
print('Euclidean similarity:', euclidean_distances(tfidf_matrix[0], tfidf_matrix[1]))

def l1_normalize(v):
    norm = np.sum(v)
    return v / norm

tfidf_norm_l1 = l1_normalize(tfidf_matrix)
print('Euclidean similarity (norm):', euclidean_distances(tfidf_norm_l1[0], tfidf_norm_l1[1]))
```

# 맨하탄 유사도

- 맨하탄 유사도(Manhattan Similarity)
  - 두 벡터 간의 거리로 유사도를 판단(기준: 맨하탄 거리판단)

```python
import numpy as np
from sklearn.feature_extraction.text import TfidfVectorizer

sent = ("휴일 인 오늘 도 서쪽 을 중심 으로 폭염 이 이어졌는데요, 내일 은 반가운 비 소식 이 있습니다.", "폭염 을 피해서 휴일 에 놀러왔다가 갑작스런 비 로 인해 망연자실 하고 있습니 다.")
tfidf_vectorizer = TfidfVectorizer()
tfidf_matrix = tfidf_vectorizer.fit_transform(sent) # document vectorization
print(tfidf_matrix)

idf = tfidf_vectorizer.idf_
print(dict(zip(tfidf_vectorizer.get_feature_names(), idf)))

# Euclidean distance
from sklearn.metrics.pairwise import manhattan_distances
print('Manhattan similarity:', manhattan_distances(tfidf_matrix[0], tfidf_matrix[1]))

def l1_normalize(v):
    norm = np.sum(v)
    return v / norm

tfidf_norm_l1 = l1_normalize(tfidf_matrix)
print('Manhattan similarity (norm):', manhattan_distances(tfidf_norm_l1[0], tfidf_norm_l1[1]))
```