Text Vectorization

텍스트를 숫자로 표현하기

- 컴퓨터는 사람이 이해할 수 있는 텍스트 데이터를 이해할 수 없습니다. 따라서 컴퓨터가 이해할 수 있는 형식인 숫자 형식으로 변환 시켜서 단어를 이해시켜 줘야 합니다.

- 일반적으로 문서는 문장으로, 문장은 단어로, 단어는 문자 또는 서브워드로 이루어져 있습니다. 보통은 텍스트를 숫자로 표현할 때 단어나 문자, 서브워드를 표현해 줍니다.

Integer Encoding

정수 인코딩과 단어 집합(단어 사전)

- 사람의 언어인 텍스트를 컴퓨터에게 인식 시켜주기 위해서 컴퓨터가 이해할 수 있는 단어 사전(Vocabulary)을 먼저 만들어 줍니다.
- 단어 집합(단어 사전)은 집합이기 때문에 중복을 허용하지 않고, 순서가 없습니다.
- 정수 인코딩은 단어 집합에 있는 단어들에게 정수를 부여하여 컴퓨터에게 알려주는 방법입니다.

Vocabulary(단어 집합) 생성 방법

A : 오늘 회식 해요?

B : 오늘 메뉴 뭐였지?

A : 오늘 곱창 회식 이래요.

C : 메뉴 곱창 밖에 없나

Vocabulary(단어 집합) 생성 방법

A : 오늘 회식 해요?

B : 오늘 메뉴 뭐였지?

A : 오늘 곱창 회식 이래요.

C : 메뉴 곱창 밖에 없나

Vocabulary(단어 집합) 생성 방법

A : 오늘 회식 해요?

B : 오늘 메뉴 뭐였지?

A : 오늘 곱창 회식 이래요.

C : 메뉴 곱창 밖에 없나

단어 집합

오늘, 회식, 메뉴, 해요, 뭐였지,
곱창, 없나, 밖에, 이래요

단어 집합 기반의 정수 인코딩

정수 부여는 보통 많이 등장한 단어가 낮은 고유 정수를 갖습니다.

단어 집합
오늘, 회식, 메뉴, 해요, 뭐였지, 곱창, 없나, 밖에, 이래요

정수
인코딩

단어 집합	고유 정수
오늘	1
회식	2
해요	3
메뉴	4
뭐였지	5
곱창	6
이래요	7
밖에	8
없나	9

단어 집합 기반의 정수 인코딩

A : 오늘 회식 해요?
B : 오늘 메뉴 뭐였지?
A : 오늘 곱창 회식 이래요.
C : 메뉴 곱창 밖에 없나

정수 인코딩

단어 집합	고유 정수
오늘	1
회식	2
해요	3
메뉴	4
뭐였지	5
곱창	6
이래요	7
밖에	8
없나	9

정수 매핑

A : [1, 2, 3]
B : [2, 4, 5]
A : [1, 6, 2, 7]
C : [4, 6, 8, 9]

Padding

- 모든 입력되는 문장의 길이는 다를 수 있습니다.
- 따라서 임의의 단어 토큰 <pad>를 추가해서 모든 인코딩된 문장의 길이를 맞춰주는 과정을 Padding이라고 합니다.
- Padding을 통해 기계에게 학습할 문장의 길이를 맞춰주면 각 단어에 대한 행렬 병렬 연산을 수행할 수 있습니다.

Post Padding

단어 집합	고유 정수
〈pad〉	0
오늘	1
회식	2
해요	3
메뉴	4
뭐였지	5
곱창	6
이래요	7
밖에	8
없나	9

정수 매핑

A : [1, 2, 3, 0]

B : [2, 4, 5, 0]

A : [1, 6, 2, 7]

C : [4, 6, 8, 9]

Pre Padding

단어 집합	고유 정수
〈pad〉	0
오늘	1
회식	2
해요	3
메뉴	4
뭐였지	5
곱창	6
이래요	7
밖에	8
없나	9

정수 매핑

A : [0, 1, 2, 3]

B : [0, 2, 4, 5]

A : [1, 6, 2, 7]

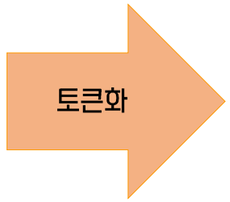
C : [4, 6, 8, 9]

OOV(Out Of Vocabulary)

- 기존 단어 집합에 존재하지 않는 단어를 Out Of Vocabulary(OOV)라고 합니다.
- 단어 집합에 없는 단어는 어떻게 정수 인코딩을 수행할 수 있을까요?
- 없는 단어라면 단어 집합에 새로운 단어로 추가 시켜도 되겠지만, 이렇게 되면 단어 집합의 크기가 너무 커져버리고 맙니다.
- 새로운 단어를 단어 집합에 추가시키지 않고, 단어 집합에 없는 단어로서 일괄적으로 하나의 토큰으로 매핑해 주는 것이 OOV 문제를 해결하는 방법입니다.

OOV 처리 과정

D : 오늘 회식 소고기



토큰화된 문장
[오늘, 회식, 소고기]

OOV 처리 과정

참고할 수 있는 정수가 없는 상태!

토큰화된 문장
[오늘, 회식, 소고기]

정수 인코딩

단어 집합	고유 정수
〈pad〉	0
오늘	1
회식	2
해요	3
메뉴	4
뭐였지	5
곱창	6
이래요	7
밖에	8
없나	9

OOV 처리 과정

단어 집합에 〈oov〉 토큰을 추가하면?

토큰화된 문장
[오늘, 회식, 소고기]

정수 인코딩

단어 집합	고유 정수
〈pad〉	0
〈oov〉	1
오늘	2
회식	3
해요	4
메뉴	5
뭐였지	6
곱창	7
이래요	8
밖에	9
없나	10

OOV 처리 과정

토큰화된 문장
[오늘, 회식, 소고기]

정수 인코딩

단어 집합	고유 정수
〈pad〉	0
〈oov〉	1
오늘	2
회식	3
해요	4
메뉴	5
뭐였지	6
곱창	7
이래요	8
밖에	9
없나	10

정수 매핑

[2, 3, 10]

BOW(Bag Of Words)

BOW란?

- BOW(Bag Of Words)는 가방 속의 단어들 이라는 뜻입니다. 즉 단어들을 가방에 하나씩 넣어 놓고 가방을 흔들면 문장을 구성하던 순서는 의미가 없어지게 되고, 오로지 단어 자체에만 집중하게 됩니다

- 무엇에 대해 집중할까요? 일반적으로는 빈도수(Frequency)와 단어가 문장에서 얼마만큼 영향을 미치는지가 대표적 입니다.

- 참고로 Word Embedding은 단어 자체를 벡터로 표현하여 단어와 단어 사이의 관계에 집중합니다.

One Hot Encoding

- 먼저 문장으로부터 단어 집합(Vocabulary)를 만들어 내고, 문장 내에 단어가 들어 있으면 1, 단어가 없으면 0으로 표기합니다.

예시

문장 1 : 호랑이 에게 물려가도 정신만 차리면 산다.

문장 2 : 호랑이 굴에 들어가야 호랑이 새끼를 잡는다.

	호랑이	굴	물다	잡다	정신	새끼	잡다	산다	들어가다	차리다
호랑이	1	0	0	0	0	0	0	0	0	0
물다	0	0	1	0	0	0	0	0	0	0
정신	0	0	0	0	1	0	0	0	0	0
처리다	0	0	0	0	0	0	0	0	0	1
산다	0	0	0	0	0	0	0	1	0	0

DTM(Document Term Matrix)

- 먼저 문장으로부터 단어 집합(Vocabulary)를 만들어 내고, 각 문장 마다 등장한 단어의 횟수를 세어줍니다.

예시

문장 1 : 호랑이 에게 물려가도 정신만 차리면 산다.

문장 2 : 호랑이 굴에 들어가야 호랑이 새끼를 잡는다.

	호랑이	굴	물다	잡다	정신	새끼	잡다	산다	들어가다	차리다
문장 1	1	0	1	0	1	0	0	1	0	1
문장 2	2	1	0	0	0	1	1	0	1	0

TF-IDF(Term Frequency - Inverse Document Frequency)

- TF IDF는 문장 내 단어의 중요도를 계산해 줍니다.
  - TF는 하나의 문장 내 단어의 등장 횟수. 즉 DTM을 의미합니다.
  - IDF는 DF의 역수로, DF는 단어가 등장한 문서의 개수를 의미 합니다.

다음 문장들을 살펴보고, 각 문장에 대해 주제가 될 수 있는 단어들을 생각해 봅시다.

문장 1 : 피카츄는 전기 포켓몬이며, 피카츄가 진화하면 라이츄가 됩니다.

문장 2 : 꼬부기는 물 포켓몬이며, 꼬부기가 진화하면 어니부기가 됩니다.

문장 3 : 파이리는 불 포켓몬이며, 파이리가 진화하면 리자드가 됩니다.

TF-IDF(Term Frequency - Inverse Document Frequency)

- TF IDF는 문장 내 단어의 중요도를 계산해 줍니다.
  - TF는 하나의 문장 내 단어의 등장 횟수. 즉 DTM을 의미합니다.
  - IDF는 DF의 역수로, DF는 단어가 등장한 문서의 개수를 의미 합니다.

다음 문장들을 살펴보고, 각 문장에 대해 주제가 될 수 있는 단어들을 생각해 봅시다.

문장 1 : 피카츄는 전기 포켓몬이며, 피카츄가 진화하면 라이츄가 됩니다. → 피카츄

문장 2 : 꼬부기는 물 포켓몬이며, 꼬부기가 진화하면 어니부기가 됩니다. → 꼬부기

문장 3 : 파이리는 불 포켓몬이며, 파이리가 진화하면 리자드가 됩니다. → 파이리

왜 포켓몬은 주제 단어가 될 수 없을까요? 모든 문장에서 포켓몬이라는 단어가 등장하기 때문에 주제 단어로 생각하지 않습니다.

TF-IDF(Term Frequency - Inverse Document Frequency)

- TF IDF는 문장 내 단어의 중요도를 계산해 줍니다.
  - TF는 하나의 문장 내 단어의 등장 횟수. 즉 DTM을 의미합니다.
  - IDF는 DF의 역수로, DF는 단어가 등장한 문서의 개수를 의미 합니다.

다음 문장들을 살펴보고, 각 문장에 대해 주제가 될 수 있는 단어들을 생각해 봅시다.

문장 1 : 피카츄는 전기 포켓몬이며, 피카츄가 진화하면 라이츄가 됩니다. → 피카츄의 TF : 2, DF : 1, TF-IDF : 2

문장 2 : 꼬부기는 물 포켓몬이며, 꼬부기가 진화하면 어니부기가 됩니다. → 꼬부기의 TF : 2, DF : 1, TF-IDF : 2

문장 3 : 파이리는 불 포켓몬이며, 파이리가 진화하면 리자드가 됩니다. → 파이리의 TF : 2, DF : 1, TF-IDF : 2

왜 포켓몬은 주제 단어가 될 수 없을까요? 모든 문장에서 포켓몬이라는 단어가 등장하기 때문에 주제 단어로 생각하지 않습니다.

TF-IDF(Term Frequency - Inverse Document Frequency)

- TF IDF는 문장 내 단어의 중요도를 계산해 줍니다.
  - TF는 하나의 문장 내 단어의 등장 횟수. 즉 DTM을 의미합니다.
  - IDF는 DF의 역수로, DF는 단어가 등장한 문서의 개수를 의미 합니다.

다음 문장들을 살펴보고, 각 문장에 대해 주제가 될 수 있는 단어들을 생각해 봅시다.

문장 1 : 피카츄는 전기 포켓몬이며, 피카츄가 진화하면 라이츄가 됩니다. → 피카츄의 TF : 2, DF : 1, TF-IDF : 2

문장 2 : 꼬부기는 물 포켓몬이며, 꼬부기가 진화하면 어니부기가 됩니다. → 꼬부기의 TF : 2, DF : 1, TF-IDF : 2

문장 3 : 파이리는 불 포켓몬이며, 파이리가 진화하면 리자드가 됩니다. → 파이리의 TF : 2, DF : 1, TF-IDF : 2

문장 1에서 포켓몬의 TF는 1, DF는 3 입니다. 따라서 포켓몬의 TF-IDF 값은 1/3으로, 피카츄에 비해 낮습니다.

BOW의 장단점

- BOW의 장점은 쉽고 빠르게 만들 수 있다는 점입니다. 또한 TF-IDF 등은 간단한 계산을 통해 문서의 특징을 잘 나타내어 전통적으로도 활용도가 높습니다.

- 단점은 문맥의 의미가 반영되지 않으며, 희소행렬 문제가 나타난다는 것에 있습니다.