# “ 자연어처리 뿌시자 ! ”

# NLP 논문 스터디😎

## - OT

2024.01.02 | 발표자 : 유하영

# 목차

# 😊 자기소개

**유하영**

**황현태**

**오원준**

## 1. 간단한 자기소개

이름/소속/나이/거주/관심분야/MBTI 등.. 자유롭게

## 2. NLP 공부여부

NLP 공부여부, 관련 프로젝트 경험여부

# 😎스터디 진행

❖ 시간

**매주 목요일 오전 11시~오후 1시** (약 **1시간 반** 정도 진행)

❖ 장소

**온라인** (Discord 화상회의) **https://discord.gg/mnQpcVyT**

❖ 목표

**논문을 통한 자연어처리 이해 및 기술 구현**

자연어처리의 기본 원리 및 최신 NLP 연구 동향을 논문을 통해 이해하고,
나아가 이론을 실제로 구현하는 것을 목표로 한다.

❖ 기간

[기초] 2개월(~3월초) -> [심화] ?

# 😎스터디 진행방식

- 매주 한 명씩 돌아가며 본인이 맡은 논문의 리뷰를 진행해요.
- **진행순서 : 논문 리뷰(20~50분) -> 현 진행상황 공유(10분) -> 마무리(5분)**
  코드 구현 현황, 어려운 점 등

- 스터디 진행 방향 : [NLP 기초 .part] -> [심화/최신 기술 .part]

## NLP 기초

| | |
|---|---|
| **n-gram** | **LSTM**/Bi-LSTM |
| **TF-IDF** | seq2seq |
| **Word2Vec** | **Attention Mechanism** |
| **GloVe** | **Transformer** |
| FastText/ELMo | **BERT** |
| RNN | GPT |

## 심화/최신 기술

가능하면 인용 수가 많고
공식 구현 코드가 있는 논문들부터,

읽고 싶은 논문을 가져와 자유롭게 리뷰

# 😎스터디 진행방식

## NLP 기초

1. Basic Embedding Model
    - NNLM
    - Word2Vec
    - FastText
2. CNN
    - TextCNN
3. RNN
    - TextRNN
    - TextLSTM
    - Bi-LSTM
4. Attention Mechanism
    - seq2seq
    - Attention Mechanism
    - Bi-LSTM with Attention
5. Transformer
    - Transformer
    - BERT

https://github.com/graykode/nlp-tutorial?tab=readme-ov-file

# 😎스터디 진행방식

[NLP 기초 .part]

❖ 이론

**모든 인원이 매주 1개의 동일한 논문 읽기**
**그 중 한 명이 대표로 발표**

**발표자** : **ppt발표 진행, 스터디 직전에 Notion 및 깃허브에 발표내용 공유**
**나머지 인원** : 스터디 시간에 질문, **스터디 종료 후 Notion에 코멘트(느낀 점 등) 남기기**

❖ 구현

자신의 발표가 끝난 직후부터 **발표자는 2주간 자신이 발표한 논문 구현**
[기초]부분은 각자 맡은 논문에 따라 구현 난이도가 천차만별이기에 일단 한 달을 기준으로 잡음.
추가적으로 구현을 원하면 구현 가능. 하지만 자신이 맡은 논문은 최대 한 달 안에 구현 마무리를 우선 목표로!

# 스터디 진행방식

## 1. [기초] part (~ 3월 초까지. 약 8주)

- 매주 모든 인원이 **동일한 논문 읽기**
- 그 중 **한 명이 대표로 발표** (돌아가며 발표 진행)
- 스터디 진행 방식

  **논문 리뷰(20~50분) → 현 진행 상황 공유(10분) → 마무리(5분)**

**발표자**

스터디 시작 전 : 선정된 논문 읽기

발표 준비 → **Notion 및 GitHub에 발표할 내용 공유**

스터디 시간 : PPT 발표 진행

스터디 종료 후 : 2주간 발표한 논문 구현 후 Git에 Push하기 + *새로운 논문 읽기*

**나머지 인원**

스터디 시작 전 : 선정된 논문 읽기

스터디 시간 : 발표 경청 후 질문

스터디 종료 후 : Notion에 **코멘트(느낀 점 등) 남기기** → *새로운 논문 읽기*

# 스터디 진행방식

**발표순서**

| 일시 | 갑 | 을 | 병 | 정 |
|---|---|---|---|---|
| 스터디 | OT 진행 | | | |
| | 논문 1 읽기 | 논문 1 읽기 | 논문 1 읽기 | 논문 1 읽기 |
| 스터디 | **논문 1 발표** | | | |
| | **논문 1 구현**<br>논문 2 읽기 | 논문 2 읽기 | 논문 2 읽기 | 논문 2 읽기 |
| 스터디 | | **논문 2 발표** | | |
| | **논문 1 구현**<br>논문 3 읽기 | **논문 2 구현**<br>논문 3 읽기 | 논문 3 읽기 | 논문 3 읽기 |
| 스터디 | 논문 1 구현 공유 | | **논문 3 발표** | |
| | 논문 4 읽기 | 논문 4 읽기 | **논문 3 구현**<br>논문 4 읽기 | 논문 4 읽기 |
| 스터디 | | 논문 2 구현 공유 | | **논문 4 발표** |

# 😎스터디 진행방식

[심화/최신 기술 .part]

❖ 이론
각자 읽고 싶은 **논문을 선정해 읽기** (각자 2주에 1개의 논문을 읽는 셈)
매주 1~2명씩 돌아가면서 발표

**발표자** : **ppt발표 진행, 스터디 직전에 Notion 및 깃허브에 발표내용 공유**
**나머지 인원** : 스터디 시간에 질문, **스터디 종료 후 Notion에 코멘트(느낀 점 등) 남기기**

❖ 구현
자신의 발표가 끝난 직후부터 **발표자는 한 달간 자신이 발표한 논문 구현 시작**

# 😎스터디 진행방식

https://github.com/jiphyeonjeon/season1/blob/main/beginners/README.md

## ❖발표 예시

https://www.youtube.com/watch?v=Hd-LctlQJ7I

**02 Methodology**
Proposed Rank Algorithm

- Boundary-Aware Centrality
  - ✓ 기존 전통적인 centrality 계산 방식은 후보 phrase 들의 상대적인 위치를 고려하지 않음
  - ✓ 하지만 일반적으로 문서 내에서 중요 내용(phrase)은 문서의 초반이나 후반 부에 등장
  - ✓ Boundary: 문서의 시작과 끝
  - ✓ 어떤 문서에 서로 다른 n 개의 후보 phrase 들 존재
  - ✓ $d_b(i)$: $i$ 번째 phrase 의 위치와 문서의 시작과 끝의 상대적인 위치 정보를 비교해줄 수 있는 boundary function

$$d_b(i) = \min(i, \alpha(n-i)) \quad \cdots (5)$$

  - ✓ $\alpha$:Boundary 의 상대적 중요성을 조절하는 hyper-parameter / $\alpha$ <1: 시작 부근의 phrase가 중요 , $\alpha$>1: 마지막 부근의 phrase 가 중요
  - ✓ $d_b(i) < d_b(j)$ 일 경우 phrase $i$ 가 phrase $j$ 보다 boundary 에 가까이 있는 것임

$$C(H_{KP_i}) = \sum_{d_b(i)<d_b(j)} e_{ij} + \lambda \sum_{d_b(i)\geq d_b(j)} e_{ij} \quad \cdots (6)$$

*$\lambda \in (0,1)$: boundary 근처에 존재하지 않은 phrase 의 영향을 줄이기 위한 hyper parameter
$\lambda \to 1$: boundary 부근의 phrase 들의 영향력이 커짐

17

**Subword Model**

3.2 Subword model

By using a distinct vector representation for each word, the skipgram model ignores the internal structure of words. In this section, we propose a different scoring function $s$, in order to take into account this information.

Each word $w$ is represented as a bag of character $n$-gram. We add special boundary symbols < and > at the beginning and end of words, allowing to distinguish prefixes and suffixes from other character sequences. We also include the word $w$ itself in the set of its $n$-grams, to learn a representation for each word (in addition to character $n$-grams). Taking the

eating ⟶ <eating>

위처럼 단어의 양 끝에 <> 를 더하여
접두사와 접미사를 구분할 수 있도록 했다

**Subword Model**

<wh, whe, her, ere, re>
and the special sequence
<where>.

Note that the sequence <her>, corresponding to the word *her* is different from the tri-gram her from the word where. In practice, we extract all the $n$-grams for $n$ greater or equal to 3 and smaller or equal to 6. This is a very simple approach, and different sets of $n$-grams could be considered, for example taking all prefixes and suffixes.

<eating>
<eating>
3-grams <ea eat ati tin ing ng>

| Word | Length(n) | Character n-grams |
|---|---|---|
| eating | 3 | <ea, eat, ati, tin, ing, ng> |
| eating | 4 | <eat, eati, atin, ting, ing> |
| eating | 5 | <eati, eatin, ating, ting> |
| eating | 6 | <eatin, eating, ating> |

3 ≤ n ≤ 6 범위의 n-gram을 사용하였다

**Subword Model**

Suppose that you are given a dictionary of $n$-grams of size $G$. Given a word $w$, let us denote by $\mathcal{G}_w \subset \{1, \ldots, G\}$ the set of $n$-grams appearing in $w$. We associate a vector representation $\mathbf{z}_g$ to each $n$-gram $g$. We represent a word by the sum of the vector representations of its $n$-grams. We thus obtain the scoring function:

$$s(w, c) = \sum_{g \in \mathcal{G}_w} \mathbf{z}_g^\top \mathbf{v}_c.$$

This simple model allows sharing the representations across words, thus allowing to learn reliable representation for rare words.

In order to bound the memory requirements of our model, we use a hashing function that maps $n$-grams to integers in 1 to $K$. We hash character sequences using the Fowler-Noll-Vo hashing function (specifically the FNV-1a variant).[1] We set $K = 2.10^6$ below. Ultimately, a word is represented by its index in the word dictionary and the set of hashed $n$-grams it contains.

단어를 n-gram 벡터의 합으로 나타냄

단어 간에 표현을 공유하도록 하여
희소 단어도 의미 있는 표현을 배움

**1. Sequence to sequence (no attention)**

**Long Term Dependency Problem**

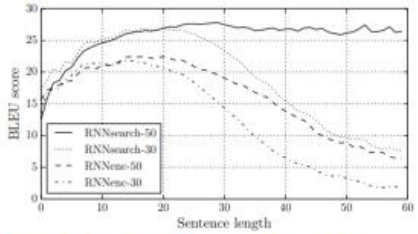



Figure 2: The BLEU scores of the generated translations on the test set with respect to the lengths of the sentences. The results are on the full test set which includes sentences having unknown words to the models.

RNNsearch : sequence to sequence model with additive attention

RNNenc : sequence to sequence model without attention (encoder-decoder model)

Trained model with sequence length up to 30 (RNNsearch-30 / RNNenc-30) or 50 (RNNsearch-50 / RNNenc-50)

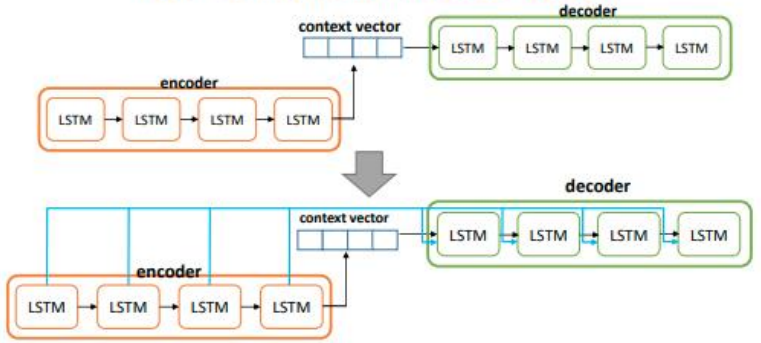
RNNsearch-50 is the best !
RNNsearch-30 > RNNenc-50

-> When dealing with long sentences, The attention mechanism can improve MT performance.

Bahdanau et al.
https://arxiv.org

**to sequence (no attention)**

**If so, how about passing all values of the encoder's hidden state cell to the input of the decoder?**




**The size of the vector you have to consider becomes very large, causing sparse problems.**

# 😎스터디 진행방식

❖ 기록

- **Notion** 및 GitHub 사용

Notion

https://www.notion.so/oneull/885f9bb984ef4c43ba43e4f51079ffe4

댓글 추가

## NLP 논문 스터디

스터디는 다음과 같은 방식으로 진행됩니다!

논문 스터디 진행 소개

[ 스터디 준비 요약 ]

LIST

### List of Papers

| Week | Key | Paper | Presenter | Date |
|---|---|---|---|---|
| 1주차 | NNLM | A Neural Probabilistic Langu | 유하영 | 2024/01/11 |
| 2주차 | Word2Vec (Ski | Distributed Representations | | 2024/01/18 |
| 3주차 | FastText | Bag of Tricks for Efficient Text C | | 2024/01/25 |
| 4주차 | TextCNN | Convolutional Neural Networks | | 2024/02/01 |

+ 새로 만들기

갯수 4

### Study Info

- 스터디 시작일: 2024.01.02 (화)
- 노션 생성일: 2024.01.02 (화)
- 스터디 장소 : Discord Link :)
- 스터디 시간: 매주 목요일 11:00
- 스터디 구성원: 유하영, 황현태, 오원준

### About Us

유하영

황현태

오원준

# 😎스터디 진행방식

❖ 기록

- **Notion 및 GitHub 사용**

### 😺 GitHub

1. [발표자] README 작성 (링크 달기)
2. [발표자] 제작한 PPT 및 코드 push

### Presentations

- 01 : Long Short-Term Memory
  - Paper, Video, Presentation
  - *S. Hochreiter, J. Schmidhuber, Neural Computation 1997*
  - Keywords: LSTM, Neural Network
  - Presentor : 송석리
- 02 : Efficient Estimation of Word Representations in Vector Space
  - Paper, Video, Presentation
  - *Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). arXiv preprint arXiv:1301.3781.*
  - Keywords: Word2Vec
  - Presentor : 이영빈