

# 단어 의미 중의성 해소

## (Word Sense Disambiguation)

월간 자연어

2018. 08. 29.

Jeff.yu@kakaocorp.com

# 목차

---

- 배경
  - 단어 중의성 해소란?
  - 선행 연구 조사
- 실험 소개
  - Overview
  - 베이스라인
  - SVM 모델
  - 딥러닝 모델
- 결론

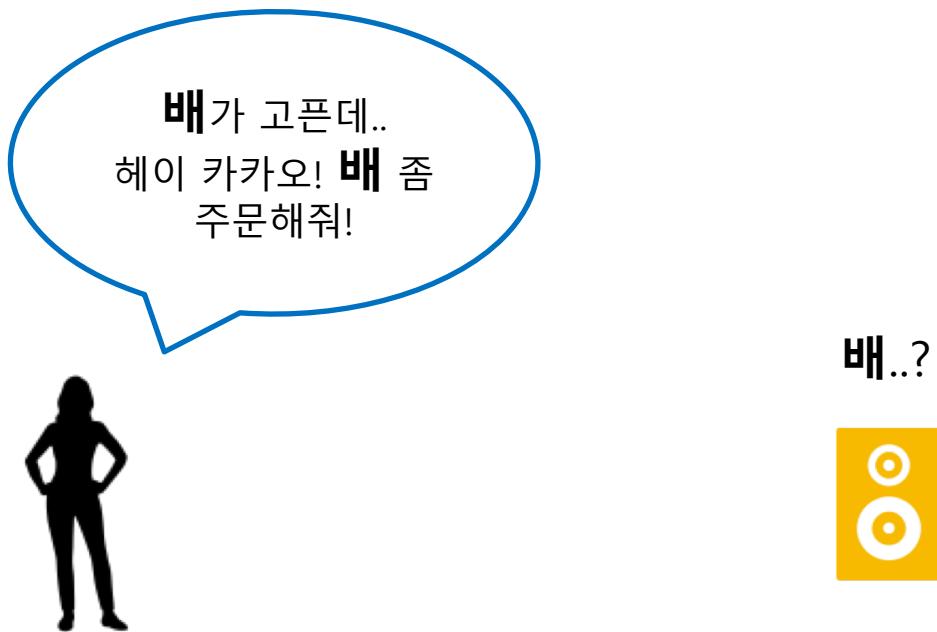
단어 의미 중의성 해소란?

# 단어 의미 중의성 해소란?

---

- 동형이의어의 의미 구분

“이처럼 똑같이 생긴 동형이의어의 의미를 구분하는 문제”



# 단어 의미 중의성 해소란?

---

“한국산 수출품 배에 대하여”



한국\_05/NNP 산\_08/XSN 수출품/NNG 배\_03/NNG 에/JKB 대하\_02/VV 여

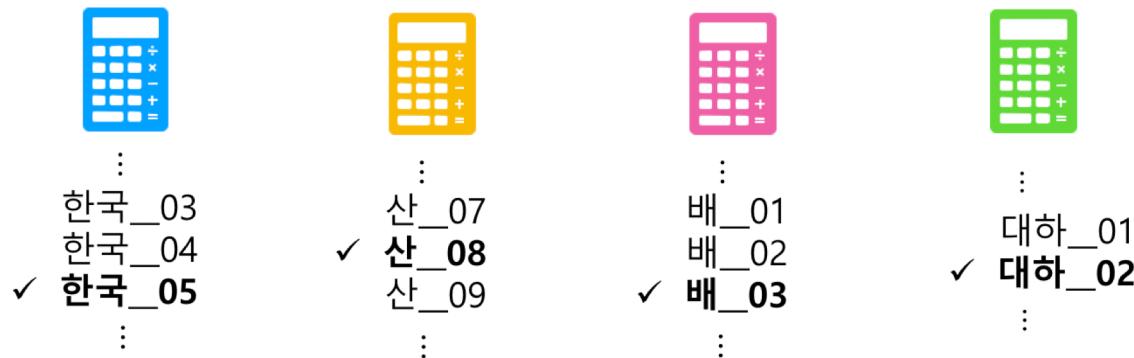
단순히 수가 많아서 어려운건 아니고

# 단어 의미 중의성 해소란?

예를들어 형태소 태깅같은 경우는 암만 단어 수가 많아도 다 똑같은 레이블, 똑같은 문제인데

- 각 동형이의어 별 분류(classification) 문제

“한국산 수출품 배에 대하여”

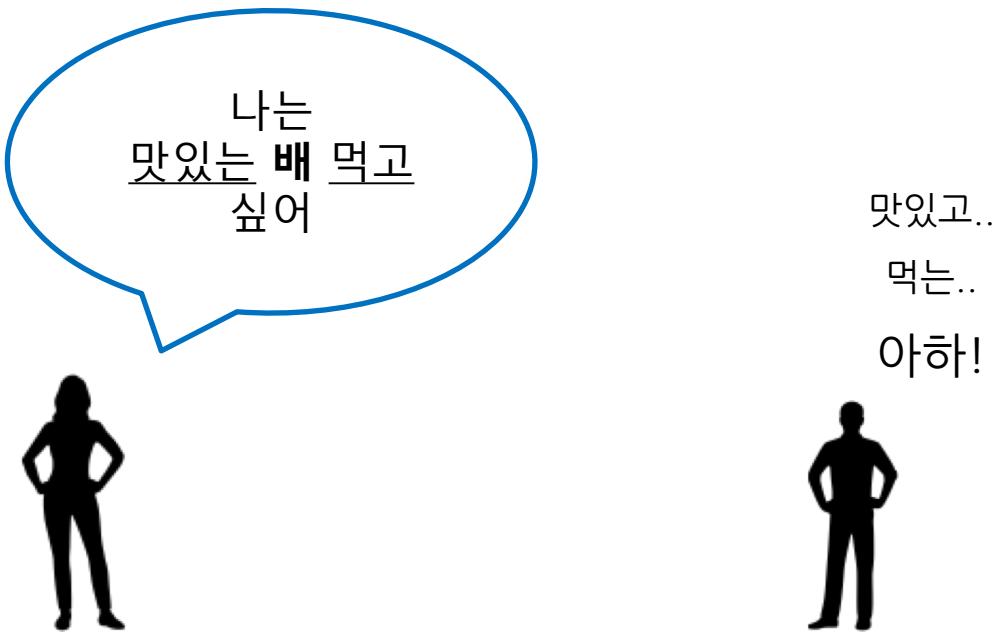


한국\_05/NNP 산\_08/XSN 수출품/NNG 배\_03/NNG 에/JKB 대하\_02/VV 여

# 단어 의미 중의성 해소란?

---

- 사람은?
- 앞뒤 문맥을 보고 단어의 의미를 유추!



그래서 보통 컴퓨터로도 사람이 하는것처럼 앞뒤 문맥을 보고 그걸 실마리로 해서 문제를 해결합니다.

# 단어 의미 중의성 해소란?

---

“앞뒤 문맥을 통해 동형이의어의 의미를 알아내는 **분류 문제**”

이제 부터 진짜 제 2달동안의 발자취를 한번 따라가 볼텐데  
제일 먼저 했던일은 선행 연구 조사였어요

## 선행 연구 조사

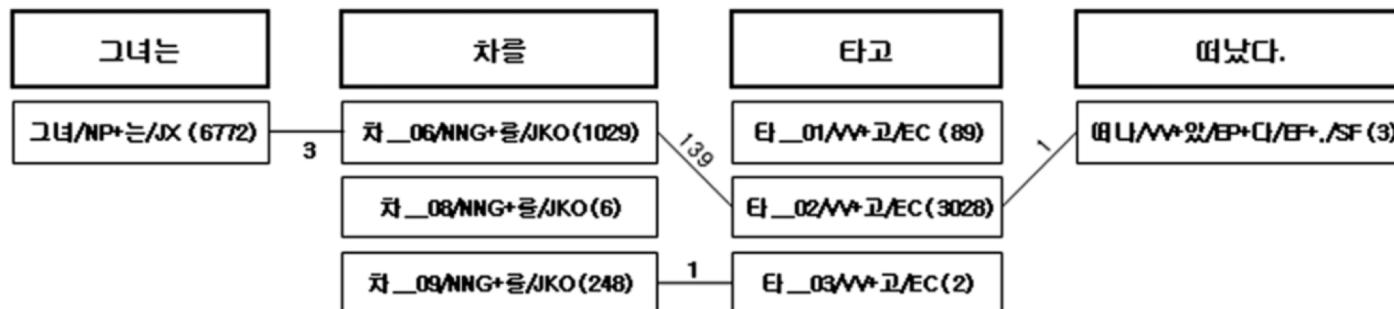
# 선행 연구 조사

---

- 공기빈도정보를 이용한 연구
- 기계학습 분류기를 이용한 연구
- 딥러닝 모델 이용한 연구

# 선행 연구 조사

- 한국어 어휘의미망(UWordMap)을 이용한 동형이의어 분별 개선  
(신준철, 옥철영, 2015)
  - 조건부 확률(공기 정보) 계산 모델 사용 (HMM)
  - UWordMap 을 통해 상위어 경로 추출 후 중의성 해소에 사용
  - 세종코퍼스 110만 어절에 대해 98.5% 정확도.



저희 실험과 같은 데이터셋인 세종코퍼스에 대해 98.5% 정확도 - 아주아주 높다  
높은 정확도에 대한 이야기는 뒤에가서 또 할꺼니까 일단 넘어가고

# 선행 연구 조사

---

- Embeddings for word sense disambiguation: An evaluation study  
(Ignacio Iacobacci, 2016)

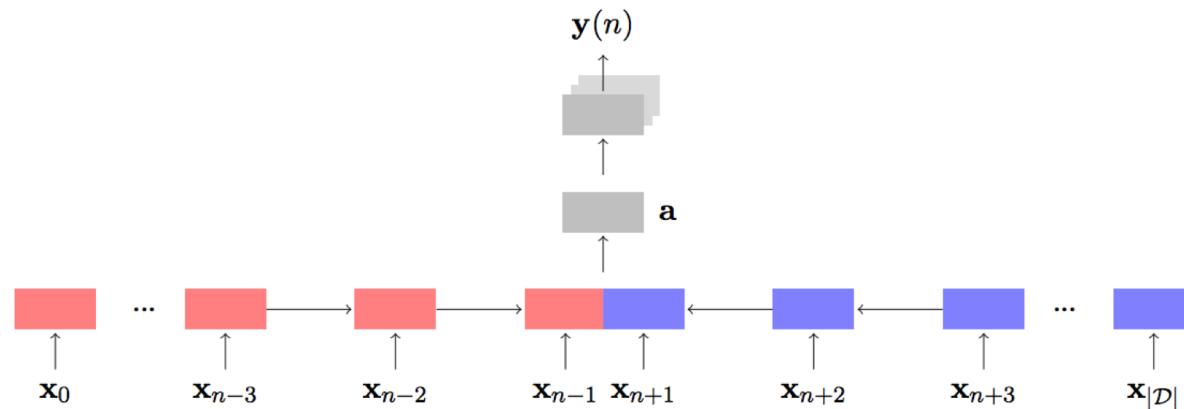
- SVM 사용
- 워드 임베딩 활용 피쳐벡터 구성
- 피쳐벡터 구성에 다양한 시도
- 딥러닝 등장 이전까지 가장 좋은 성능

| System                   | SE2         | SE3         | SE7         |
|--------------------------|-------------|-------------|-------------|
| MFS baseline             | 60.1        | 62.3        | 51.4        |
| IMS (Zhong and Ng, 2010) | 68.2        | 67.6        | 58.3        |
| Taghipour and Ng (2015)  | —           | <b>68.2</b> | —           |
| IMS (pre-trained models) | 67.7        | 67.5        | 58.0        |
| IMS (SemCor)             | 62.5        | 65.0        | 56.5        |
| IMS (OMSTI)              | 67.0        | 66.4        | 57.6        |
| IMS + Word2vec (SemCor)  | 63.4        | 65.3        | 57.8        |
| IMS + Word2vec (OMSTI)   | <b>68.3</b> | <b>68.2</b> | <b>59.1</b> |

Table 3: F1 performance on different English all-words WSD datasets.

# 선행 연구 조사

- Word Sense Disambiguation using a Bidirectional LSTM.  
(Mikael Kageb ° ack, 2016)
  - LSTM을 사용한 딥러닝 모델
  - 좌우 문맥정보를 이용할 수 있다는 장점 활용



all word 정확도는 찾아보기 힘들었는데 그래도 특정 조건에서 뛰어난 성능

정리하자면~~3개가 대표적인 접근 방법이었고 또한가지 말씀드릴것은 한국어 연구는 특히 딥러닝쪽이나 분류기 쪽은 많지않았고 있어도 정확도를 구한 데이터셋이 일관된 기준이 잘 없어서 조금 참고하기가 어려웠어서 이렇게 영어 논문으로 대신 설명을 드렸고 저는 세가지 중에 svm이랑 딥러닝 참고해서 실험 진행했습니다

# 실험 소개

# 실험 소개

---

- 데이터 셋 : 세종 코퍼스 기반 울산 코퍼스
- 트레이닝 셋 9 : 테스트 셋 1
- 형태소 분석까지 되어있는 말뭉치**

| 분류   | 문장 수    | 어절 수      | 형태소 수      | 동형이의어 수 |
|------|---------|-----------|------------|---------|
| 트레이닝 | 846,931 | 9,945,890 | 21,515,393 | 23,526  |
| 테스트  | 94,103  | 1,100,559 | 2,393,077  | 15,219  |

형태소 분석이 된 말뭉치 = 품사구분, 활용신경 안써도됨 => 난이도가 조금 쉬워졌다

한국/NNP 산/XSN 수출품/NNG 배/NNG 예/JKB 대하/VV 여



한국\_05/NNP 산\_08/XSN 수출품/NNG 배\_03/NNG 예/JKB 대하\_01/VV 여

# 베이스라인

---

서사적으로 설명 : 가장 처음 한일은 베이스라인 측정입니다.,.

- 트레이닝 셋에서 모든 동형이의어의 빈도 정보를 수집.
- 테스트 시 각 동형이의어에 대해 **가장 자주 등장한 의미번호**  
(Most Frequent Sense)를 태깅

쉽게 말해서 앞뒤 다 안보고 가장 자주 등장한 의미로 무조건 태깅하겠다, 약간 무식한 방법이죠?

**배/NNG** 를 타다

배\_01/NNG : 38회

배\_02/NNG : 233회

배\_03/NNG : 2회



**배\_02/NNG** 를 타다

- 정확도 : 약 96%, Too high!!

# 실험 방향 설정

---

어링 뭔가 이상하다. 분석을 해봐야겠다

- 베이스라인이 너무 높다
- 일단 단어별 난이도를 나누어 구해보자!
- Shannon 엔트로피
  - $$H = - \sum_{i=1}^N p(x_i) \cdot \log p(x_i)$$
  - 어떤 분포가 있을때 그 분포의 불안정한 정도를 나타내는 수치
  - 분포가 **넓고 고를수록** 엔트로피 **High**
  - 분포가 **좁고 몰려있을수록** 엔트로피 **Low**
  - 엔트로피를 단어별 의미 중의성 해소 난이도 척도로 이용.

# 실험 방향 설정

```
def entropy(count_list):
    return sum(map(lambda y: -y/sum(count_list)*np.log2(y/sum(count_list)), count_list))

a = [1000, 1, 1, 1, 1, 1, 1]      # 쉬운 분포
b = [50, 50, 40, 45, 30, 49, 23] # 어려운 분포

ea = entropy(a)
eb = entropy(b)
print("ea = {}".format(ea))
print("eb = {}".format(eb))

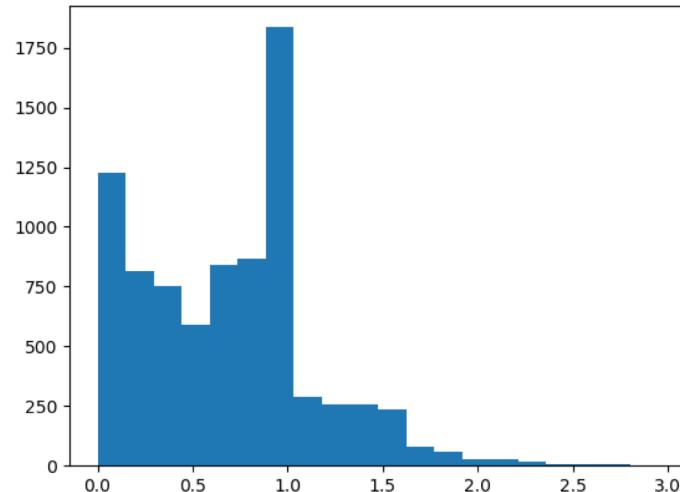
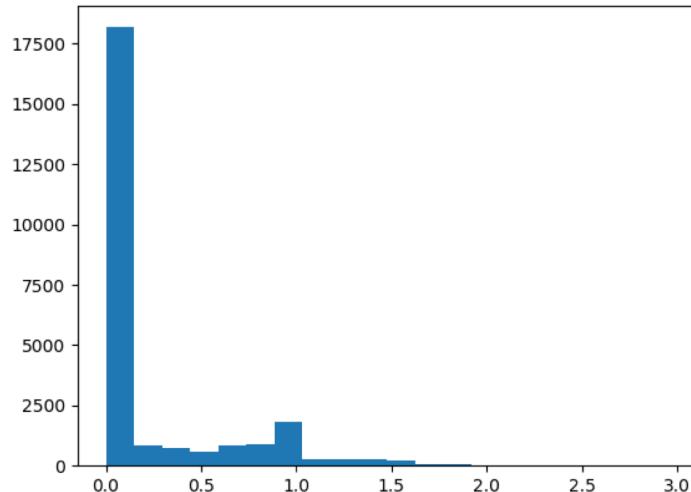
ea = 0.06806838238794549
eb = 2.7615426154317353
```

```
word: 열리 /VV, entropy: 0.1, distribution: [4209, 50]
word: 증세 /NNG, entropy: 0.1, distribution: [318, 4]
word: 소형 /NNG, entropy: 0.1, distribution: [233, 3]

word: 감수 /NNG, entropy: 2.5, distribution: [6, 6, 9, 2, 1, 1, 1, 1]
word: 원 /NNG, entropy: 2.5, distribution: [119, 151, 9, 45, 52, 2, 20, 1, 24, 5, 1]
word: 관 /NNG, entropy: 2.5, distribution: [138, 88, 2, 66, 2, 3, 200, 10, 18, 1, 7, 5, 2, 1, 2, 1]
```

# 실험 방향 설정

그래서 이 엔트로피를 트레이닝셋 전체에 나타난 동형이의어 들에 대해 다 측정해봤어요



엔트로피가 0: 난이도가 제로, MFS로 하면 100% 정답

- 총 23526개의 동형이의어 중 엔트로피가 0인 동형이의어 수가 17046개(**70%**)
- 엔트로피가 0 근처인 동형이의어들 : 난이도가 **아주 쉽고 아주 많다**. MFS로만 처리해도 충분.
- 엔트로피가 0.1~1.5 : 난이도가 **평이한 대신 수가 많다**.
- 엔트로피가 2.0 이상 : 난이도가 **어려운 대신 수가 적다**.

# 실험 방향 설정

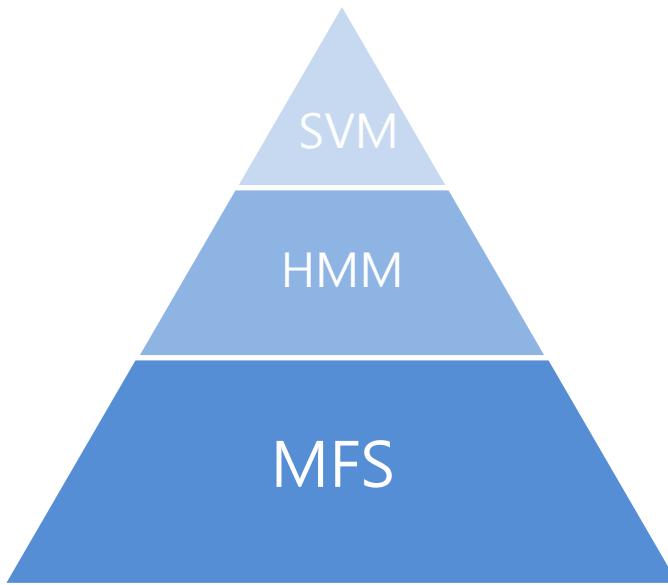
---

아 데이터셋 엔트로피 분포가 이렇구나 그렇다면

“엔트로피별로 모델을 달리하여 문제를 해결해보자!”

# 실험 방향 설정

---



- **엔트로피별로 모델을 달리하여 문제를 해결해보자!**
  - 엔트로피가 0 근처인 동형이의어들 : MFS
  - 엔트로피가 0.1~ 1.5 : 공기정보기반 확률계산 모델
  - 엔트로피가 최상위 : SVM 분류기 모델

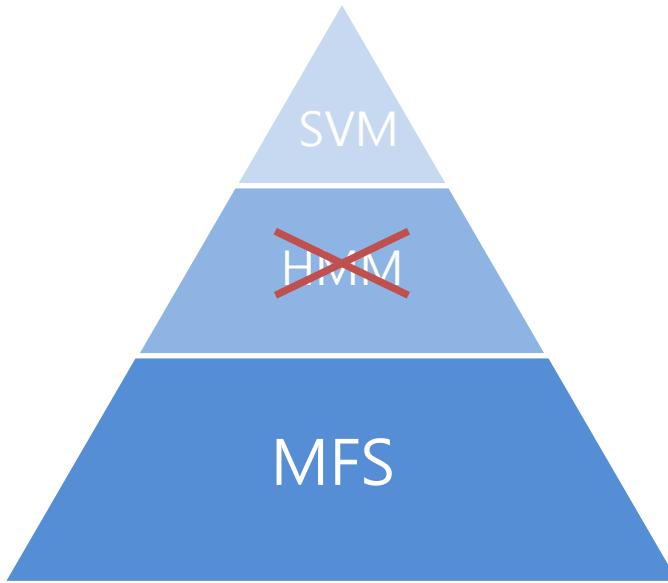
# 실험 방향 설정

---

- 초기 계획은 SVM으로 높은 난이도의 수 백개의 단어에 대해서만 처리하고자 함.
- **왜?**
  - WSD = 각각의 동형이의어에 대한 서로 다른 분류 문제
  - 각각의 동형이의어에 대해 각각 다른 분류기가 필요
  - 각각의 분류기를 다 따로 학습시켜야 함.
- **그런데...?**
  - 생각보다 각각의 동형이의어에 대한 학습 인스턴스가 많지 않음 (평균 200개의 학습 인스턴스)
  - 각각의 분류기를 학습하는데 그리 오랜 시간이 걸리지 않음
- **그래서!**
  - 그냥 엔트로피 0.1 이상 동형이의어에 대해 각각 다 SVM 모델을 만들어 학습해봄
  - 총 6109 개의 동형이의어, 6109개의 SVM 모델

# 실험 방향 설정

---



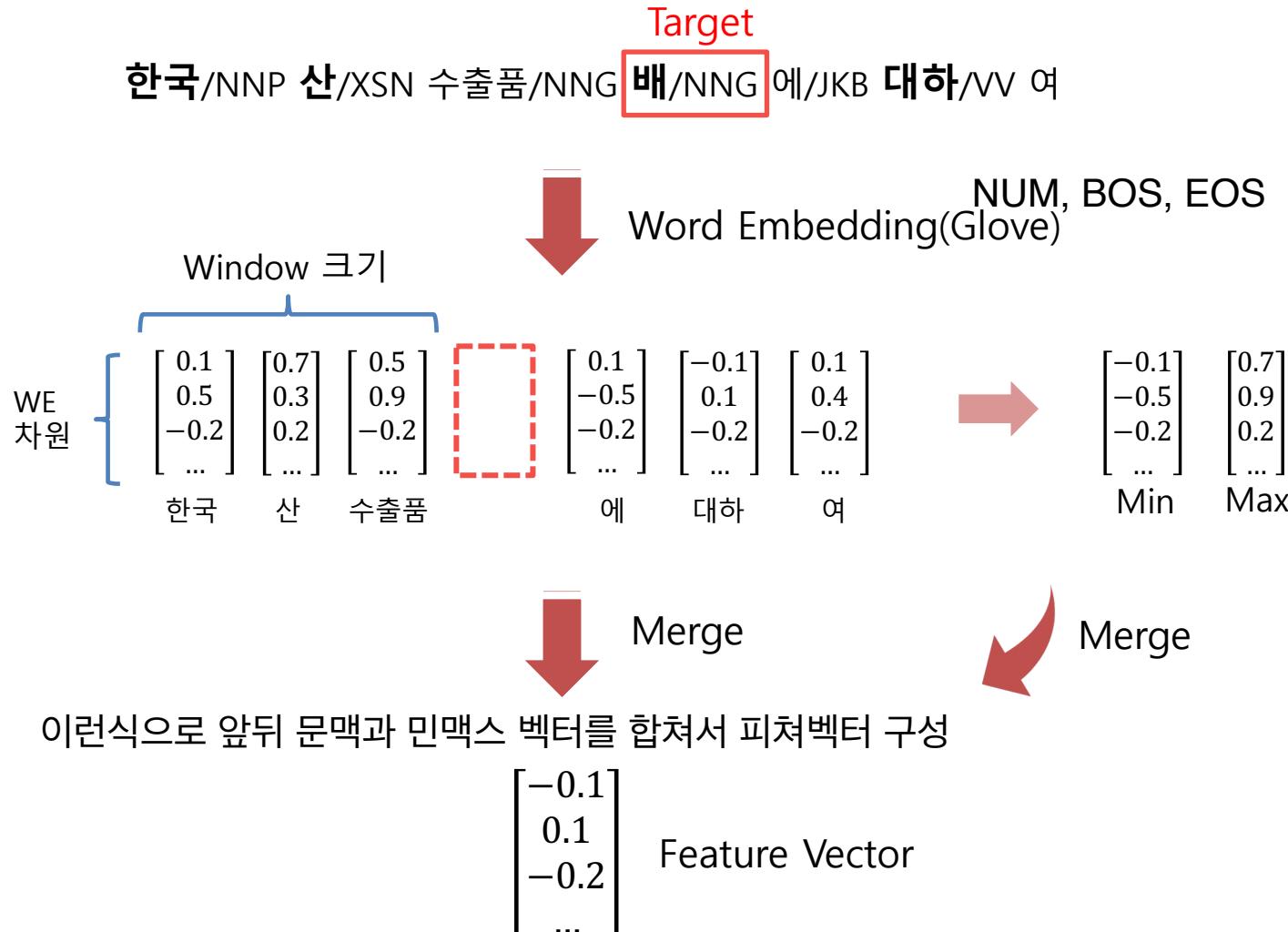
- 중간 난이도까지 SVM 모델로 커버해버리자!
  - 엔트로피가 0 근처인 동형이의어들 : MFS
  - 엔트로피가 0.1~ 1.5 : ~~공기정보기반 확률계산 모델~~ → SVM 분류기 모델
  - 엔트로피가 최상위 : SVM 분류기 모델

정리: 그래서 일단 어려운, 중간 난이도 단어는 svm, 쉬운단어는 MFS로 해보자

# SVM 모델

# SVM 모델 - 피쳐벡터

제가 직접 SVM 모델의 세부 수식이나 커널 이런걸 건드릴만큼은 안되서 그런건 라이브러리의 도움을 받고  
피쳐벡터만 어떻게 잘 구성해보자 해서 설명드릴게 피쳐벡터 밖에 없는데



# SVM 모델 - 결과

---

앞서 말했던 중간 나이도와 높은 나이도의 모든 동형이의어에 대해서

- 엔트로피 0.1 이상 동형이의어 대상, 총 6109개의 SVM 모델을 각각 학습
- 다양한 파라미터에 대해 실험
  - 워드 임베딩 방법 : **Glove**, Word2Vec
  - 윈도우 사이즈 : 2, 5, 10
  - 임베딩 차원 : 10, 50, 100, 200
  - Merge 방법 : Sum, **Concat**
  - 추가 feature : **Min-Max vector**

| min<br>max | Merge         | win | dim | accuracy           |
|------------|---------------|-----|-----|--------------------|
| T          | <b>concat</b> | 5   | 200 | <b>93.06676964</b> |
| T          | concat        | 2   | 200 | 92.9102015         |
| T          | concat        | 5   | 100 | 92.88913852        |
| F          | concat        | 5   | 200 | 92.76135646        |
| T          | concat        | 10  | 200 | 92.74099558        |
| T          | concat        | 2   | 100 | 92.65955206        |
| T          | concat        | 10  | 100 | 92.64199958        |
| F          | concat        | 5   | 100 | 92.52966369        |

- SVM + MFS 모델
  - 베이스라인보다 약 2.5% 상승
  - 선행 연구중 best 모델과 비슷한 수준

| Entropy    | Model          | Accuracy      |
|------------|----------------|---------------|
| 0.1이상      | SVM            | 93.0 %        |
| <b>all</b> | <b>SVM+MFS</b> | <b>98.5 %</b> |
| all        | 선행 연구(HMM)     | 98.5%         |
| all        | 베이스라인(MFS)     | 96 %          |

# 다음단계 구상

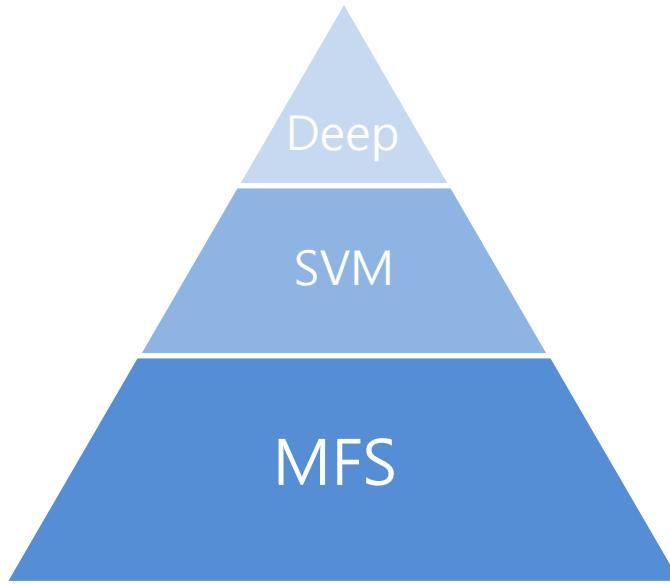
---

사실은 여기서 끝낼수도 있어요, 그냥 모든단어에 대해 svm 만들어서 학습시켰더니 정확도 잘 나오더라~  
하지만 그러면 아름답지 않으므로

- 사실 원래는 가장 어려운 단계에 적용하고자 했던 SVM 모델이 중간 난이도까지 커버
- 초기에 SVM 으로 풀려고 했던 가장 높은 난이도의 문제를 딥러닝으로 풀어보자!  
왜? 딥러닝이니까~ 더 잘나오겠지~ 딥러닝에대한 근거없는 신뢰
- **딥러닝 모델이 SVM 모델 보다 더 어려운 단어에 어울릴 수 있겠다는 직관**

# 다음단계 구상

---

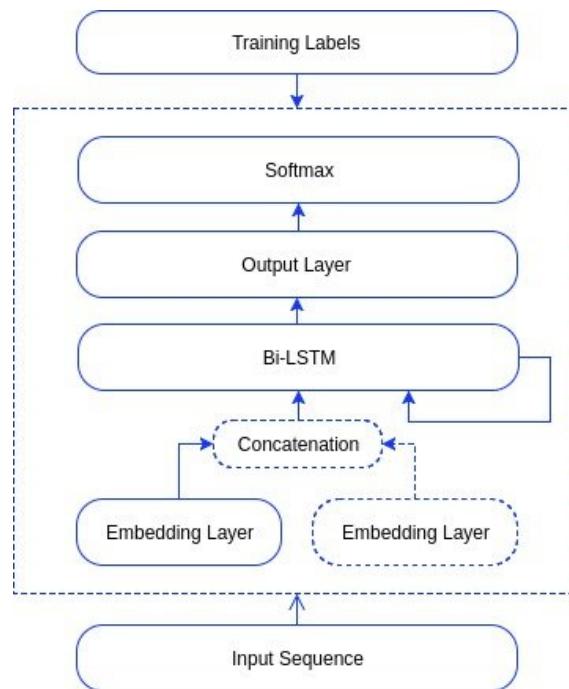


- 최상위 난이도 동형이의어는 딥러닝으로 풀어보자!
  - 엔트로피가 0 근처인 동형이의어들 : MFS
  - 엔트로피가 0.1~ : SVM 분류기 모델
  - 엔트로피 최상위: 딥러닝 모델

# 딥러닝 모델

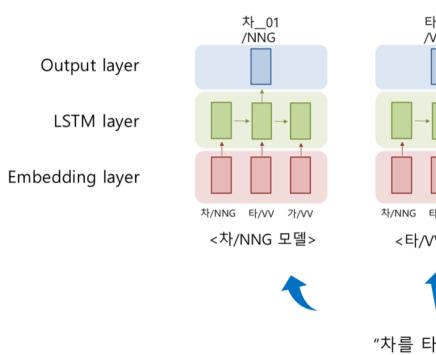
# 딥러닝 모델 - 구조

- 대부분의 선행연구에서 보편적으로 취하고 있던 모델 구조 채택
- Embedding layer - Bi-LSTM layer - Fully connected output layer

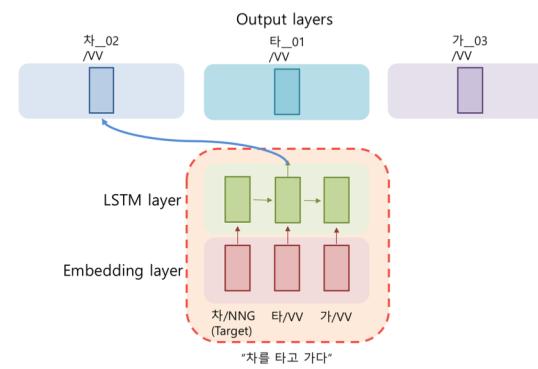


# 딥러닝 모델

- 두 가지 모델에 대해 실험



<개별 모델>



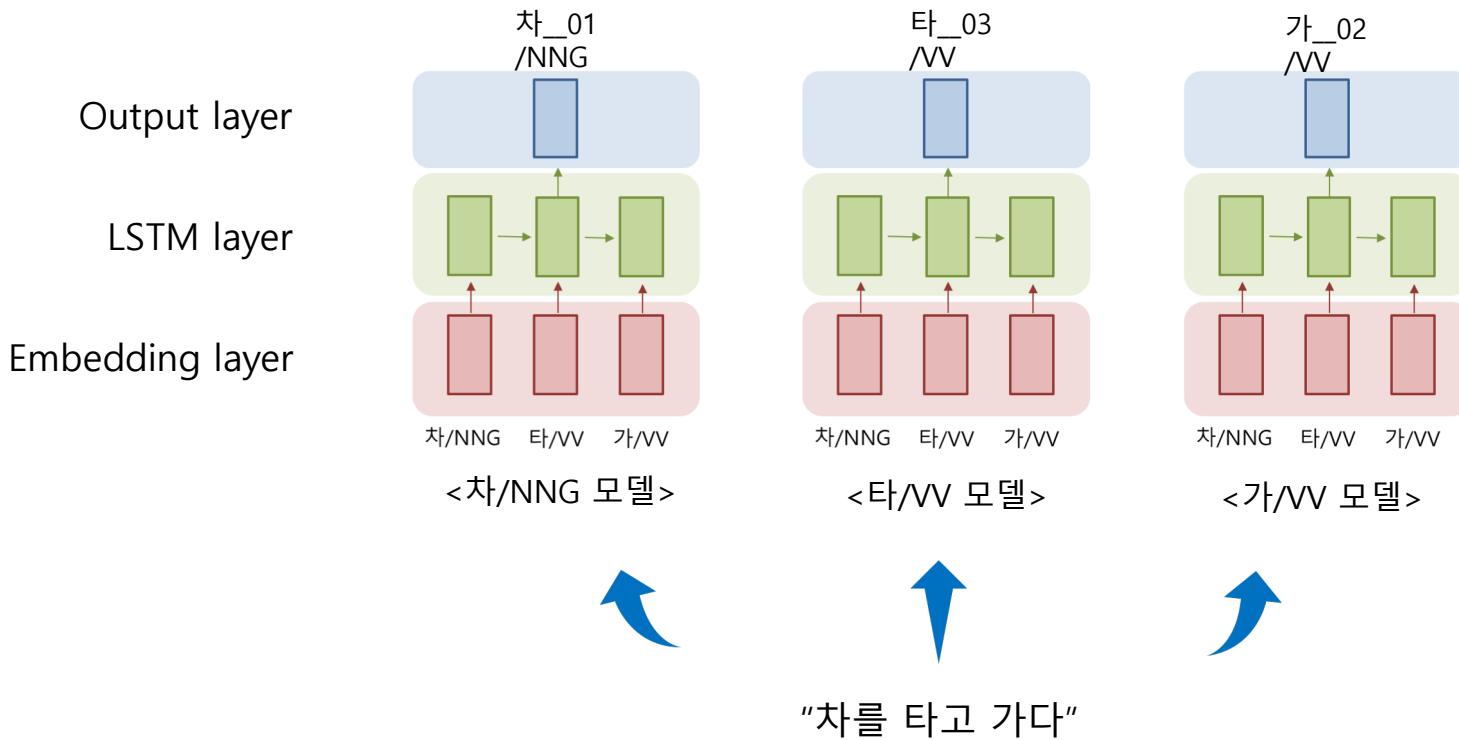
<공유 모델>

그림만 보고는 잘 이해가 안되실 테니까 개별모델 먼저 설명을 드리면

# 딥러닝 모델 - 개별 모델

- **개별 모델**

쉽게 말해서 SVM으로 했던걸 딥러닝으로 바꿔서 한거. 무식하게



# 딥러닝 모델 - 개별 모델

---

- 우선 SVM에서 했듯 각각의 동형이의어에 대해 서로 다른 모델을 만들어서 학습하여 정확도를 측정
- 임의로 선택한 9개 단어에 대한 정확도 비교

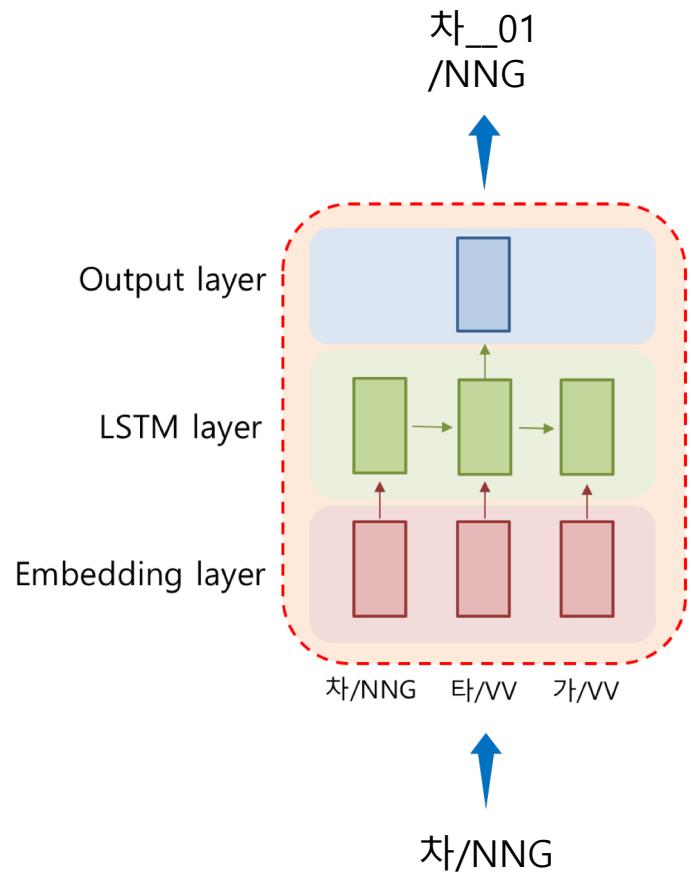
| 모델            | /NN<br>G | 원<br>/NNG | 감수<br>/NNG | 정수<br>/NNG | 거사<br>/NNG | 유<br>/NNP | 국<br>/NNG | 표시<br>/NNG | 배출되<br>/VV |
|---------------|----------|-----------|------------|------------|------------|-----------|-----------|------------|------------|
| 엔트로피          | 2.5      | 2.5       | 2.5        | 2.0        | 2.0        | 1.5       | 1.5       | 1.0        | 1.0        |
| TestCase<br>수 | 65       | 47        | 2          | 12         | 2          | 53        | 54        | 70         | 4          |
| 개별            | 84.61    | 87.23     | 0.00       | 75.00      | 50.00      | 79.24     | 90.74     | 91.42      | 100.00     |
| SVM           | 75.38    | 85.10     | 0.00       | 83.33      | 100.00     | 84.90     | 92.59     | 82.85      | 100.00     |

# 딥러닝 모델 - 이슈1

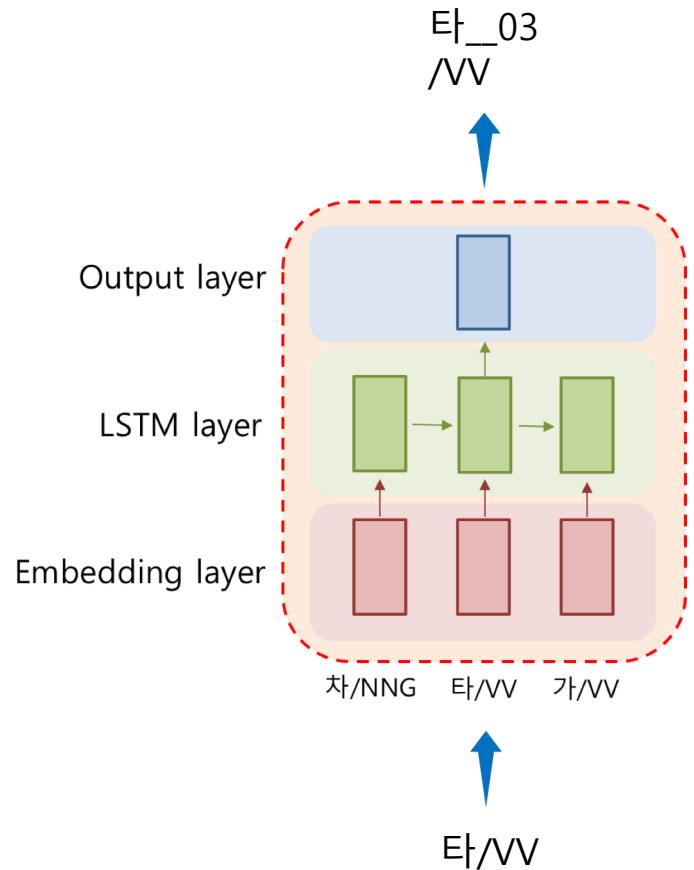
---

- 몇몇 동형이의어에 대해 SVM 보다 뛰어난 성능.
- 하지만.. SVM 정도는 ok 그냥 6000벌 만들자 할 수 있지만 딥러닝은 정말
  - 딥러닝 모델을 SVM처럼 수많은 단어들에 대해 각각 모델을 만들고 학습하기에는 비효율적
  - 또한 하나의 동형이의어에 대한 학습 인스턴스 수가 충분하지 않음
- 그렇다면 무식하게 6000벌 모델 만들지 말고
  - 하나의 모델로 모든 동형이의어의 의미를 분류해낼 수 없을까?

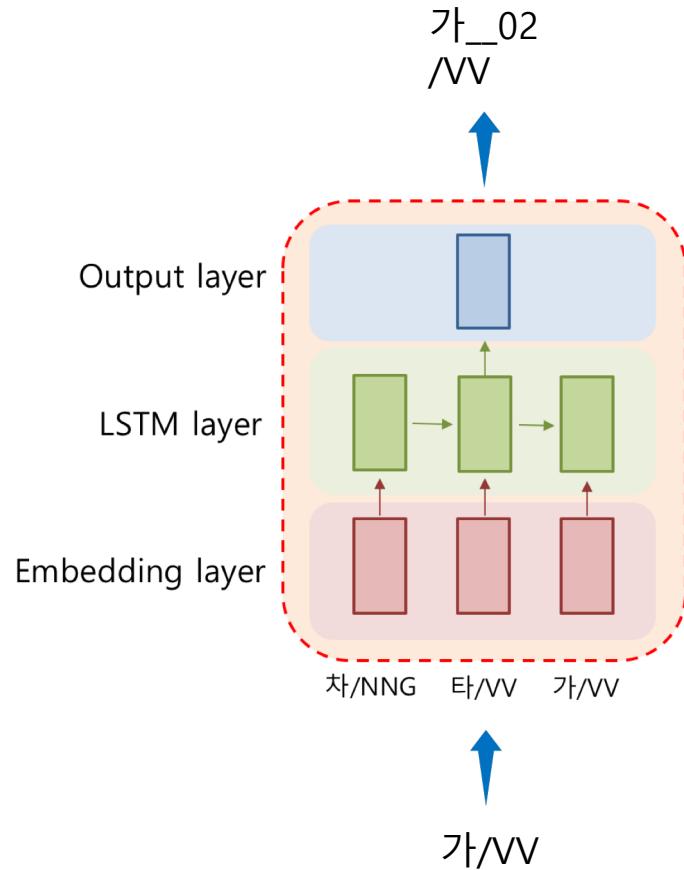
# 딥러닝 모델 - 공유 모델 구상



# 딥러닝 모델 - 공유 모델 구상



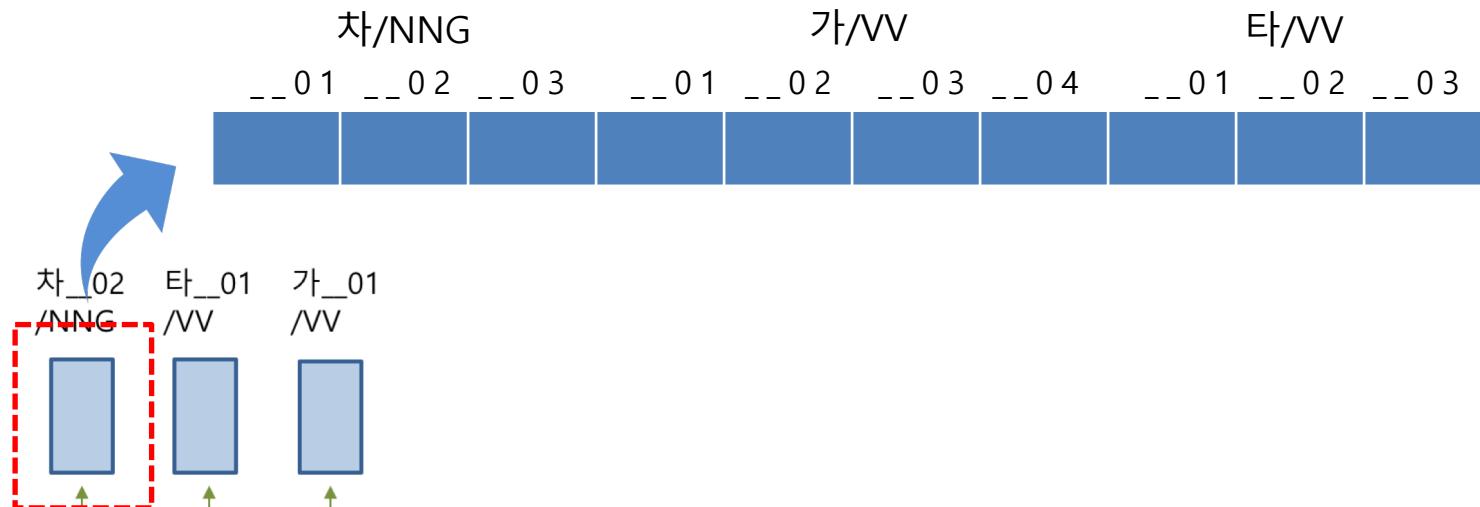
# 딥러닝 모델 - 공유 모델 구상



이렇게하면 1벌만 있어도 모든 동형이의어에 대해 처리할 수 있을텐데. 아름답

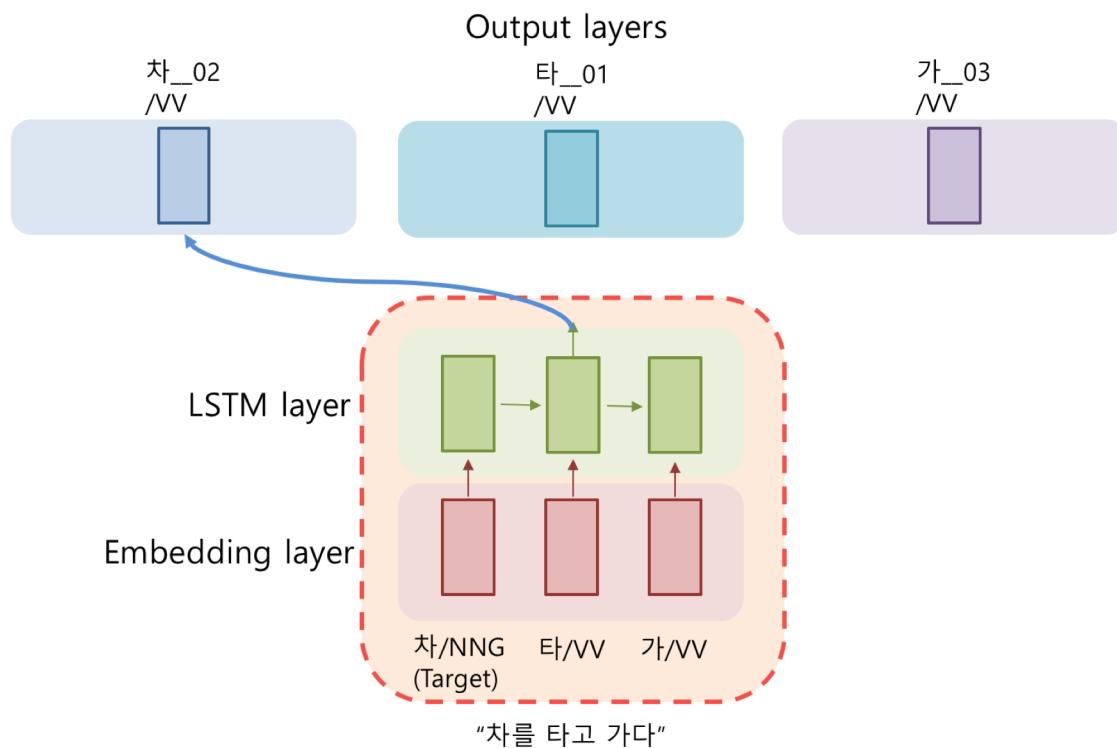
# 딥러닝 모델 - 이슈2

- **하나의 output layer**      일반적인 분류문제의 아웃풋 차원은 분류 레이블 갯수
  - 하나의 아웃풋 레이어가 모든 동형이의어의 의미를 분류해 낼 수 있어야함.
  - 즉, 아웃풋 레이어의 차원이 모든 동형이의어의 의미 갯수를 다 합친 차원이 되어야함!
  - 엔트로피가 0.1 이상인 동형이의어 수는 6000여개, 평균 의미갯수는 2.5개,  
이를 다 커버하려면 아웃풋 레이어의 차원은 15000 차원이 되어야함.
  - 평균 2.5개 중에 하나 고르는 분류 문제를 15000개 중에 하나 고르는 분류 문제로 어렵게 만들어서 풀어야 하는셈



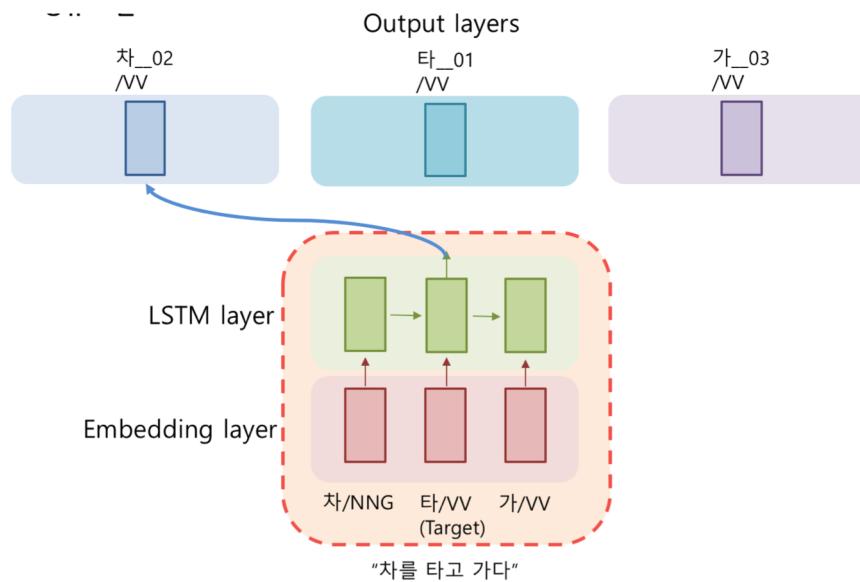
# 딥러닝 모델 - 공유 모델 구상

- 공유 모델



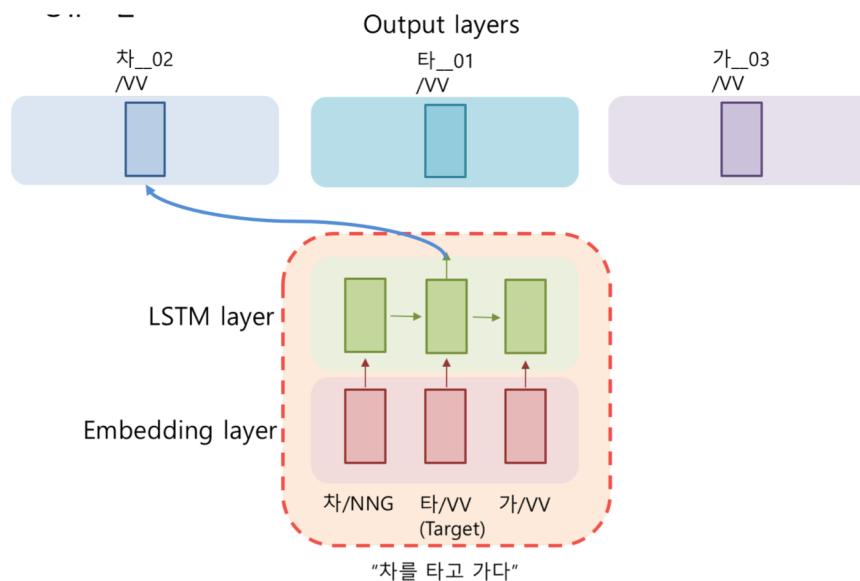
# 딥러닝 모델 - 공유 모델 구상

- 공유 모델
  - 하단 레이어들을 공유
  - 동형이의어 마다 각자의 아웃풋 레이어를 가짐
  - 하단레이어는 공유 학습 + 아웃풋 레이어는 각자 학습



# 딥러닝 모델 - 공유 모델 구상

- 공유 모델에 대한 직관, 가정
  - 타겟 동형이의어가 무엇이냐에 관계없이 **비슷한 풀이법을 공유할 것**
  - 학습 인스턴스가 적어서, 개별로는 **공통의 풀이법 조차 학습하기 힘든** 동형이의어들을 보완  
왜냐면 아랫부분은 모든 인스턴스가 통과하며 학습하니까



뭔가 그럴싸한것 같아서 약간 설렘을 가지고 실험을 했습니다

# 딥러닝 모델 - 공유 모델 결과

- 9개 샘플 단어와 나이도 상위 N개 단어에 대해 실험
- 9단어에 대한 정확도 비교

| 모델         | 관 /NNG       | 원 /NNG       | 감수 /NNG | 정수 /NNG      | 거사 /NNG       | 유 /NNP       | 국 /NNG       | 표시 /NNG      | 배출되 /VV |
|------------|--------------|--------------|---------|--------------|---------------|--------------|--------------|--------------|---------|
| 엔트로피       | 2.5          | 2.5          | 2.5     | 2.0          | 2.0           | 1.5          | 1.5          | 1.0          | 1.0     |
| TestCase 수 | 65           | 47           | 2       | 12           | 2             | 53           | 54           | 70           | 4       |
| 공유         | 81.53        | 76.59        | 0.00    | 75.00        | 50.00         | 73.58        | 85.18        | 88.57        | 100.00  |
| 개별         | <b>84.61</b> | <b>87.23</b> | 0.00    | 75.00        | 50.00         | 79.24        | 90.74        | <b>91.42</b> | 100.00  |
| SVM        | 75.38        | 85.10        | 0.00    | <b>83.33</b> | <b>100.00</b> | <b>84.90</b> | <b>92.59</b> | 82.85        | 100.00  |



- 나이도 상위 N단어에 대한 정확도 비교

| 모델  | N=10         | 20           | 100          | 200          | 500          |
|-----|--------------|--------------|--------------|--------------|--------------|
| 공유  | 75.78        | 77.42        | 80.16        | 79.77        | 83.22        |
| SVM | <b>77.60</b> | <b>79.40</b> | <b>83.37</b> | <b>83.89</b> | <b>84.72</b> |



# 딥러닝 모델 - 공유 모델 결과

---

- Not Good.. 무엇이 문제일까?
  - 공동의 풀이법조차 학습할 수 없을 만큼 학습 인스턴스가 적은 동형이의어에 대한 보완이 당초 공유모델의 컨셉
  - 생각보다 많은 동형이의어들이 각자 학습하기에 충분한 학습 인스턴스를 가지고 있었던게 아닐까?
  - 동형이의어별 평균 학습 인스턴스 수 : 200개
- 그렇다면,  
학습 인스턴스가 평균 보다 적은 동형이의어들만 모아서 공유모델을 학습시켜보자!

# 딥러닝 모델 - 공유 모델 결과

---

- Not Good.. 무엇이 문제일까?
  - 공동의 풀이법조차 학습할 수 없을 만큼 학습 인스턴스가 적은 동형이의어에 대한 보완이 당초 공유모델의 컨셉
  - 생각보다 많은 동형이의어들이 각자 학습하기에 충분한 학습 인스턴스를 가지고 있었던게 아닐까?
  - 동형이의어별 평균 학습 인스턴스 수 : 200개
- 그렇다면,  
학습 인스턴스가 평균 보다 적은 동형이의어들만 모아서 공유모델을 학습시켜보자!
- 난이도 상위 500동형이의어 중, 총 학습 인스턴스 수가 N보다 적은 동형이의어만 모아서 학습 결과

(N: 학습 인스턴스 수)

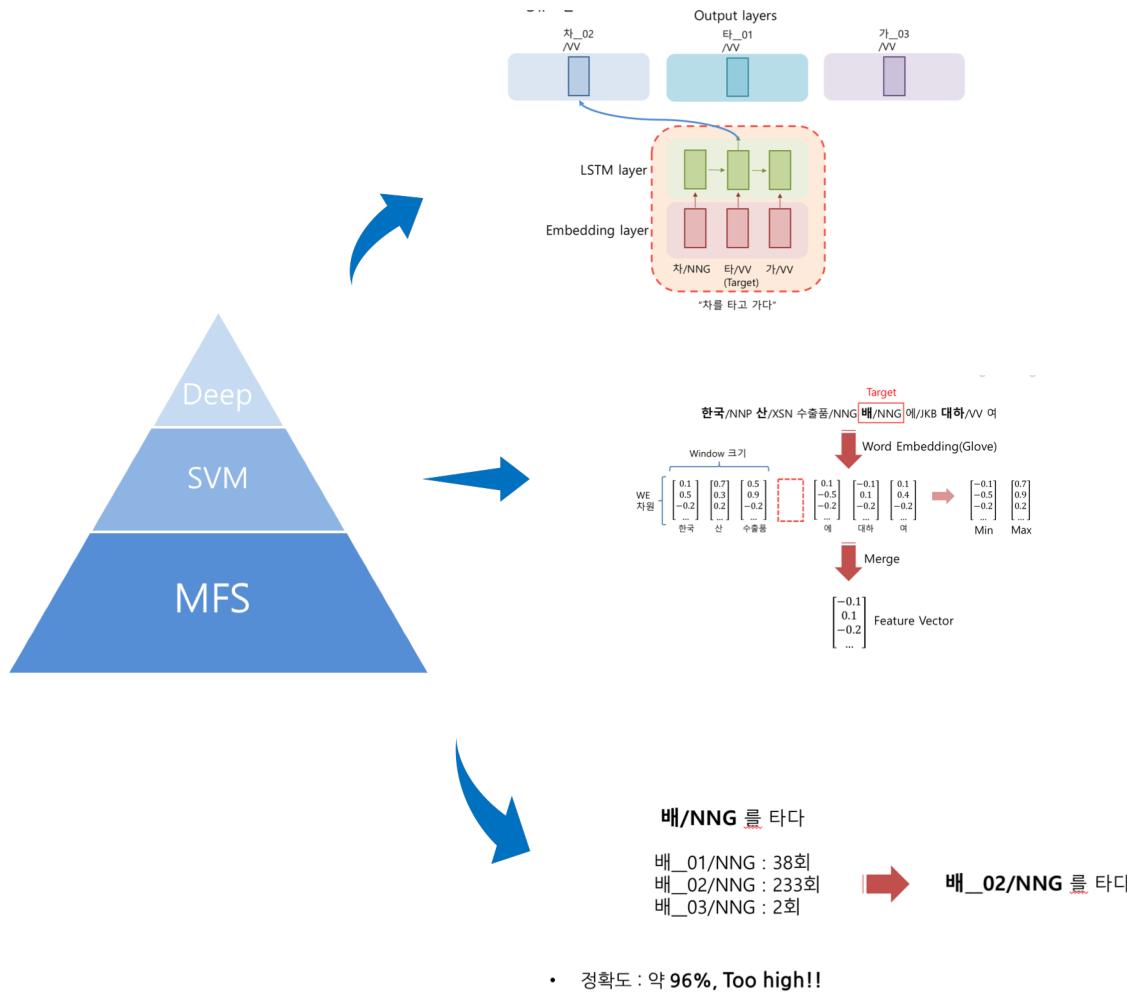
| 모델      | N=20         | 30           | 50  |
|---------|--------------|--------------|---|
| 동형이의어 수 | 117          | 148          | 204   |
| 공유      | 47.23        | 48.92        | 56.70  |
| SVM     | <b>55.27</b> | <b>58.57</b> | <b>61.68</b>  |

# 딥러닝 모델 - 공유 모델 최종 결과

---

SVM > 공유모델

# 결론



# 감사합니다

월간 자연어

2018. 08. 29.

Jeff.yu@kakaocorp.com