# 한국어 단어 중의성 해소 문제에 대한 SVM 분류기의 정확도 분석 및 딥러닝 모델 제시

(Accuracy Analysis of SVM Classifier and DNN Model Proposal for Korean WSD)

지도교수 : 유승주

이 논문을 공학학사 학위 논문으로 제출함.

2018년 12월 27일

서울대학교 공과대학
컴 퓨 터 공 학 부
유 근 국

2019년 2월

## 초 록

이 연구는 한국어 단어 중의성 해소 문제에 대해 SVM 분류기와 딥러닝 모델을 사용하여 그 성능을 알아보고자 하였다. 실험 말뭉치로는 세종말뭉치를 사용하였으며 실질적인 정확도 분석을 위해 섀넌 엔트로피를 기준으로 중의성 단어별 난이도를 구분하였다. SVM 분류기의 정확도 분석을 위해서 다양한 자질벡터를 구성, 비교해보았으며 구성항목은 워드 임베딩 방법, 윈도우 사이즈, 임베딩 차원, 병합방법, Min-Max 벡터 여부이다. 딥러닝 모델은 Output Layer를 교체하는 방식을 제시하였으며 난이도 상위 500단어 및 학습인스턴스가 적은 단어들에 대해 정확도를 측정해보았다. 실험결과 SVM 분류기의 성능은 선행연구중 가장 뛰어난 성능을 보인 연구에 필적하는 성능을 보였으나 세종말뭉치의 베이스라인 성능이 높다는 사실은 한계점으로 보인다. 딥러닝 모델의 성능은 SVM 분류기의 성능에 미치지 못했으나 적은 학습 데이터로도 일정수준 이상의 성능을 보였다는것에 의미가 있다.



## 키 워 드

단어 중의성 해소, WSD, SVM 분류기, 딥러닝, 세종말뭉치, 섀넌 엔트로피


# 1. 서론

최근 IT 업계의 가장 큰 화두는 단연 기계학습을 필두로 한 '인공지능'이라고 할 수 있다. 2016년 봄, 알파고가 보여준 인간과의 승부는 많은 사람들에게 인공지능과 인간이 함께 양립하는 세상이 더 이상 영화 속이나 먼 미래의 이야기가 아니라 눈앞에 닥친 현실임을 깨닫게 해주었다.

자연어 처리 기술은 이러한 인공지능과 인간의 양립이라는 거대한 변화에서 열쇠가 되는 기술 중 하나이다. 음성 혹은 텍스트로 이루어진 인간의 언어를 인공지능이 이해할 수 있도록 만드는, 말 그대로 인공지능과 말이 통할 수 있도록 만드는 기술이 자연어 처리기술이기 때문이다. 최근 다양한 기업에서 경쟁적으로 출시하고 있는 인공지능 스피커가 대표적인 자연어 처리 기술의 예시라고 할 수 있다.

이러한 자연어 처리 기술 중에서도 단어 중의성 해소(Word Sense Disambiguation: WSD) 문제는 동형이의어와 같이 한 단어가 여러 의미를 가질 때, 그 단어가 텍스트 내에서 어떤 의미로 사용되었는지를 식별해 내는 문제이다. 예를 들어 "배를 먹다"의 '배'는 먹는 배를 뜻하고 "배가 아프다"의 '배'는 신체 부위중 하나인 배를 뜻한다는 것을 컴퓨터가 식별해 낼 수 있도록 하는 것이라고 할 수 있다.

단어 중의성 해소를 위해서 여러 가지 기법들이 시도되어왔으나 크게 구분하면 클러스터링과 같은 비지도학습 기법과 텍스트 범주화와 같은 지도학습 기법으로 나눌 수 있다. 일반적으로 비지도학습 보다 지도학습 기법이 더욱 뛰어난 정확도를 보이지만 단어 중의성 해소문제에 지도학습을 적용하기 위해서는 중의성 단어마다 의미태그가 붙은 방대한 분량의 말뭉치가 필요하다는 문제점이 있다.

한국어 단어 중의성 해소 말뭉치는 수작업으로 의미태그를 붙여야 한다는 이유 때문에 그 규모가 굉장히 열악한 것이 사실이었으나 국립국어원 21세기 세종계획을 통해 만들어진 세종말뭉치의 등장으로 지도학습을 통한 단어 중의성 해소에 다양한 시도가 가능해졌다.

따라서 본 연구에서는 단어 중의성 해소를 위해 의미가 태깅된 세종말뭉치를 사용하였으며, 지도학습 방법 중, 지지 벡터 기계(Support Vector Machines: SVM) 분류기 방식과 심층 신경망 모델(Deep Neural Network Model) 방식을 통해 각각 학습하고, 한국어 중의성 단어에 대한 의미 분류 성능을 확인해보고자 하였다. 특히 SVM 분류기 방식에 대해선 자질벡터를 다양한 방식으로 구성해보고 성능을 비교해보았으며, 심층 신경망 모델 방식에 대해선 실험적인 모델을 제시함으로써 한국어 단어 중의성 해소 문제에 대한 심층 신경망 모델 방식의 가능성을 보이고자 하였다.

## 2. 선행연구

한국어 단어 중의성 해소는 세종말뭉치라는 방대한 분량의 말뭉치가 생기기 전과 후의 연구 양상이 크게 달라졌다. 세종말뭉치가 생기기 이전에는 비지도학습 및 사전과 같은 지식기반 연구가 주로 진행되었다.

지식기반 방법은 의미 부착 말뭉치가 없는 경우, 어휘 부족 문제를 해결하기위해 사전, 시소러스와 같은 외부 자원에서 정보를 얻는 방식이다. 대표적으로 사전 뜻풀이 말에서 추출한 의미정보에 기반한 방식이나[1], 시소러스를 이용하여 동형이의어를 분별하는 방식[2] 등이 연구되었다. 이 외에도 [3]과 같이 사전 뜻풀이를 비지도학습에 적용하는 등 의미부착 말뭉치가 없다는 한계를 극복하고자 다양한 연구가 시도되었다.

세종말뭉치가 등장한 이후에는 이를 활용한 지도학습 방식의 연구가 활발히 진행되었다. [4][5]는 지도학습에 지식기반 방식을 결합하여 정확도를 높이고자 시도하였으며, [6]은 부분어절의 조건부확률을 계산하여 동형이의어의 의미를 분별하고자 하였다. [7]은 SVM 분류기를 단어 중의성 해소문제에 적용하고자 시도하였으며 [8]은 분류의 단서가 되는 문맥벡터를 구성하는데 있어 워드임베딩을 사용하였다. 또한 [9]는 문맥벡터의 문맥 크기를 가변적으로 바꾸어 정확도를 높이고자 시도하기도 하였다.

이처럼 한국어 단어 중의성 해소 관련 연구는 세종말뭉치를 활용한 지도학습 방식의 연구가 활발히 진행되었으나 세종말뭉치를 제외하면 널리 알려진 의미부착 말뭉치를 찾아보기 힘들 정도로 데이터가 부족하고, 그에 따라 연구들 또한 한정적이라는 문제점이 있다. 뿐만 아니라 연구별로 정확도를 측정한 단어 기준 등이 일정하지 않아 연구별 비교가 쉽지 않은 실정이다. 마지막으로 심층 신경망을 단어 중의성 해소 문제에 적용하려는 시도가 활발히 이루어지지 않은 점 또한 아쉽다고 할 수 있다.

## 3. 단어 중의성 해소 실험 설계

### 1) 문제인식

단어 중의성 해소 문제는 기본적으로 분류(Classification) 문제이다. 분류 문제란 주어진 데이터를 정해진 카테고리에 따라 분류하는 문제를 말하는데, 단어 중의성 해소 문제는 주어진 중의성 단어의 여러 가지 의미중 하나의 의미로 분류하는 분류문제라고 할 수 있다. 따라서 문제해결을 위해 SVM과 같은 분류기모델을 사용하거나 분류문제를 위한 딥러닝모델을 사용

할 수 있다.

그런데 여기서 중요한 점은 이러한 분류문제가 중의성 단어 하나하나마다 서로 다른 분류문제라는 점이다. 표준국어대사전에 따르면 '사과'는 8개의 동형이의어를 가지며 '배'는 13개의 동형이의어를 가진다. '사과'의 의미를 분류하는 문제는 '사과_01', '사과_02' ... '사과_08' 중 하나의 의미를 고르는 문제가 될 것이다. 이때 '배_01', '배_02' 와 같은 '배'의 의미는 '사과' 의미 분류의 선택지에 포함할 필요가 없다. 따라서 각각의 중의성 단어들은 각각 서로 다른 분류문제를 가지게 된다.

중의성 단어들이 각각 서로 다른 분류문제를 가진다는 것은 기계학습 관점에서 보면 중의성 단어 각각에 대해서 서로 다른 분류기를 학습 시켜야 한다는 것을 의미한다. 말뭉치에 등장하는 중의성 단어의 수가 2만개 이상임을 감안할 때, 학습시켜야 할 분류기 수가 2만개 이상이라는 사실은 부담스러울 수 있다. 따라서 본 연구에서는 컴퓨팅 자원이 허용하는 만큼 SVM 분류기의 경우 모든 중의성 단어들에 대해 서로 다른 분류기를 학습하도록 하였으며 딥러닝 모델의 경우 모든 중의성 단어들에 대해 서로 다른 모델을 학습하는 것은 무리가 있다고 판단, 후술할 변칙적인 모델을 구성하여 사용하였다.

### 2) 엔트로피

본격적인 실험에 앞서 트레이닝셋에 가장 자주 나타난 의미로만 의미를 선택하는 방식인 최빈 의미 선택 방식(Most Frequent Sense: MFS)으로 세종말뭉치의 Baseline을 측정해본 결과 트레이닝셋 : 테스트셋 = 9:1 환경에서 무려 96%의 정확도를 보였다. 이는 세종말뭉치 전체적인 단어 중의성 해소 문제 난이도가 높지 않음을 의미하고, 말뭉치에 나타나는 모든 중의성 단어들에 대해 머신러닝 기법을 이용한 중의성 해소를 시도하는 것은 비효율적임을 의미한다. 따라서 본 연구에서는 세종말뭉치에 나타나는 중의성 단어들을 난이도별로 구분하고, 일정난이도 이상의 중의성 단어들에 대해서만 SVM 분류기와 딥러닝모델을 활용하여 중의성 해소를 진행하였다.

중의성 단어의 난이도를 결정하는 기준으로는 섀넌엔트로피(Shannon Entropy)를 사용하였다. 섀넌엔트로피는 정보이론에서 해당 분포가 얼마나 불확실성이 높은지를 나타내는 값으로 식 (1)과 같이 계산할 수 있다.

$$H = -\sum_{i=1}^{N} p(x_i) \log p(x_i) \qquad (1)$$

섀넌엔트로피가 높을수록 말뭉치에 더 다양한 수의 의미가 더 비슷한 빈도로 나타나는 중의성 단어라고 할 수 있고, 따라서 난이도가 높다고 할 수 있다. 예를 들어 표1의 '증세/NNG'의 경우 말뭉치에 등장하는 의미가 2개뿐이고, 그 출현빈도도 첫 번째 의미가 두 번째 의미보다 훨씬 자주 등장하므로 엔트로피가 약 0.1에 불과하고, 중의성해소 난이도도 매우 낮은편이라고 할 수 있다. 반면 표1의 '원/NNG'의 경우 말뭉치에 등장하는 의미 수가 11개로 매우 다양하고 그 출현빈도도 고른 편이므로 엔트로피가 약 2.5로 매우 높은 난이도를 보여준다고 할 수 있다.

| 단어 | 의미 번호 | 출현 빈도 | 엔트로피 |
|---|---|---|---|
| 증세/NNG | 01 | 318 | 0.1 |
| | 02 | 4 | |
| 원/NNG | 01 | 119 | 2.5 |
| | 02 | 151 | |
| | 03 | 9 | |
| | 04 | 45 | |
| | 05 | 52 | |
| | 06 | 2 | |
| | 07 | 20 | |
| | 08 | 1 | |
| | 09 | 24 | |
| | 10 | 5 | |
| | 11 | 1 | |

**표 1** 중의성단어의 의미 수 및 출현빈도에 따른 엔트로피 차이

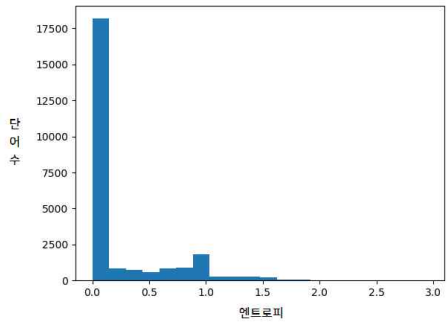


**그림 1** 세종말뭉치 중의성단어 엔트로피 분포

그림1은 세종말뭉치에 등장하는 모든 중의성 단어의 엔트로피를 측정하여 그 분포를 히스토그램으로 나타낸 것이다. 세종말뭉치에 나타난 총 23526개의 중의성 단어 중 저난이도 단어군이라고 할 수 있는 엔트로피 0.1 미만의 단어 수는 17417개로, 전체의 약 70%에 달하는 것으로 나타났다. 따라서 본 실험에서는 좀 더 실질적인 중의성해소 정확도를 구하기 위해, 이와 같은 저난이도 단어군을 제외한 엔트로피 0.1 이상의 6109개의 단어에 대해서만 머신러닝 기법을 이용하여 중의성해소를 진행, 정확도를 구해보고자 하였다.

### 3) SVM 분류기

SVM 분류기는 다양한 종류의 분류기들 중 가장 좋은 성능을 보여주는 것으로 평가되고 있다[10]. 때문에 본 실험에서는 중의성 해소를 위한 분류기 모델로 SVM을 선택하여 그 정확도를 알아보고자 하였다.

SVM은 크게 선형 SVM과 비선형 SVM으로 나뉘고 내부의 커널함수 등 다양한 파라미터에 따라 결과 및 성능이 달라진다. 하지만 본 실험에서는 이러한 SVM의 파라미터에 따른 정확도 차이를 확인하는 것이 목표가 아니므로 SVM의 세부 파라미터들은 라이브러리가 제공하는 기본값을 사용하였다. SVM 라이브러리로는 파이썬의 사이킷런(Scikit-learn) 라이브러리를 사용하였다.

SVM 내부의 세부 파라미터들 보다 분류 성능에 더 큰 영향을 미치는 것은 분류기의 입력으로 들어가는 자질벡터(Feature Vector)이다. 이러한 자질벡터를 어떻게 구성하느냐가 사실은 분류기의 분류성능을 결정한다고도 할 수 있다. 따라서 본 실험에서는 자질벡터를 다양한 방식으로 구성해보고, 가장 높은 정확도를 보여주는 자질벡터 구성방식을 찾아보고자 하였다.

그림2는 단어 중의성 해소 문제에서 자질벡터를 구성하는 방식을 표현한 도식이다. 일반적으로 중의성 해소 대상이 되는 단어의 앞뒤 문맥이 중의성해소의 가장 큰 단서가 된다. 따라서 자질벡터를 구성함에 있어서도 일정 크기의 문맥 윈도우 크기를 설정하고 문맥에 등장한 단어들을 벡터화(Word Embedding) 및 병합(Merge)하여 자질벡터를 구성하게 된다. 본 실험에서는 이와 같은 과정에서 Word Embedding 방식, 문맥 윈도우 크기, Word Embedding 차원,

Merge 방법, Min-Max vector 추가여부 등을 달리해가며 정확도를 측정하였다 (표2).

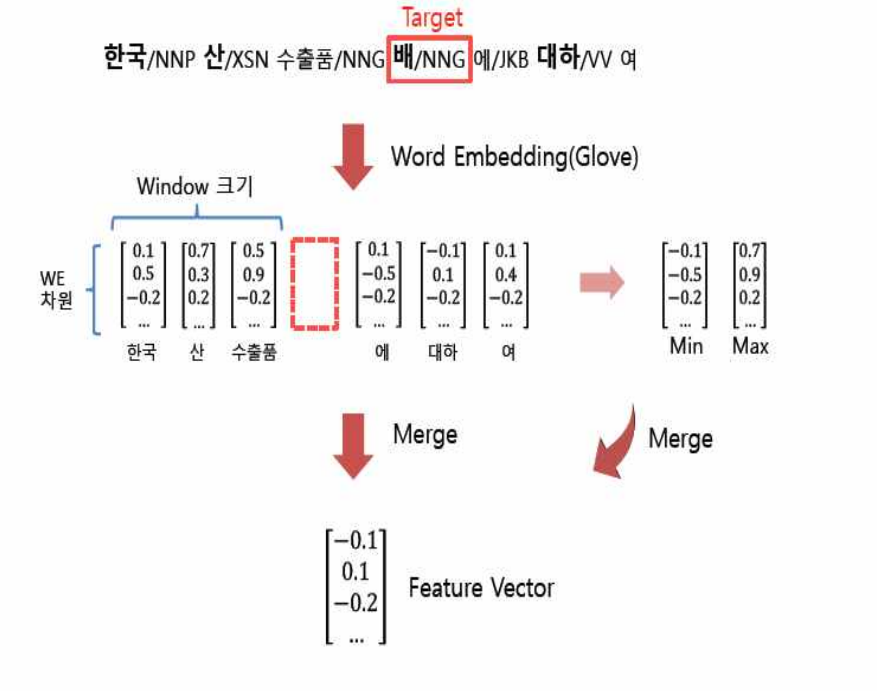


**그림 2** 자질벡터를 구성하는 과정

| 항목 | 설명 | 실험값 |
|---|---|---|
| 워드임베딩 방법 | 단어를 벡터화 하는 방법을 결정 | Word2vec, Glove |
| 윈도우 사이즈 | 중의성해소 대상 단어의 앞뒤로 몇 개까지의 단어를 문맥으로 사용할것인지를 결정 | 2, 5, 10 |
| 임베딩 차원 | 워드임베딩시 벡터화된 단어의 벡터 차원을 결정 | 10, 50, 100, 200 |
| 병합 방법 | 문맥내 단어 각각의 벡터들을 어떤식으로 병합하여 자질벡터로 만들것인지 결정 | Sum, Concat |
| Min-Max 벡터 | 문맥내 단어 각각의 벡터들에서 차원별로 최대값을 모은 벡터와 최소값을 모은 벡터를 자질벡터에 사용할지 여부를 결정 | O, X |

**표 2** 자질벡터 구성을 위한 항목 및 항목별 실험값

### 4) 딥러닝 모델

딥러닝 모델의 경우 그림3과 같은 구조로 레이어를 구성하였다. 입력 문장이 주어지면 미리 학습된 Glove Word Embedding Layer를 거쳐 문장의 모든 단어들을 벡터화 하고, 이렇게 벡터화된 Sequence가 Bi-LSTM Layer의 입력으로 들어가게 된다. 이후 Fully Connected Layer를 거쳐 최종적으로 Output Layer를 통과하면 대상단어의 의미갯수 차원의 결과벡터를 얻게된다.

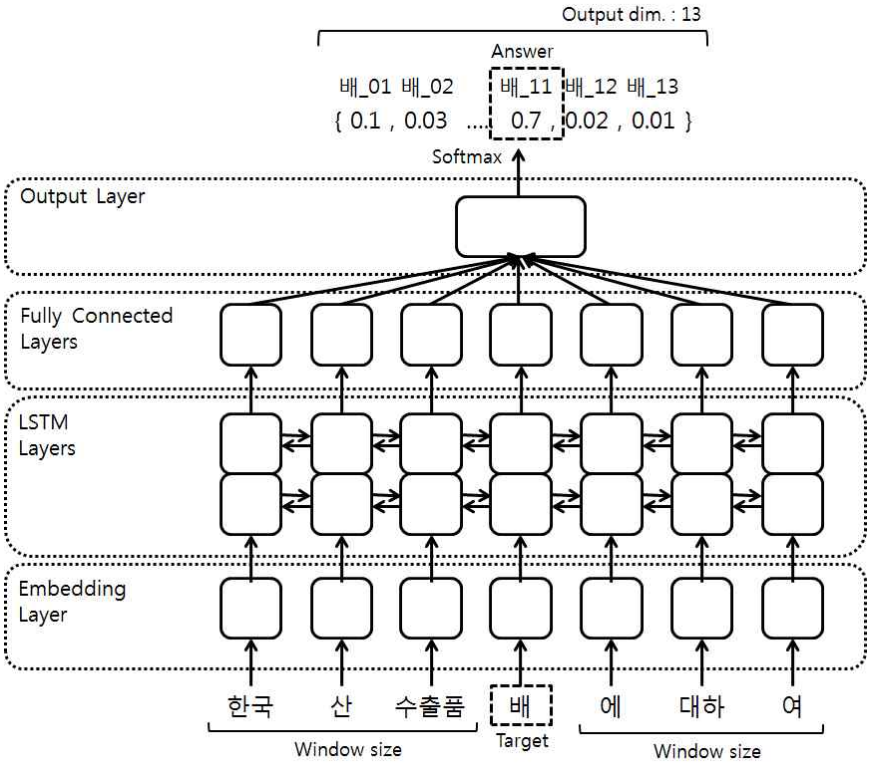


**그림 3** 딥러닝 모델 레이어 구조

그런데 여기서 다시한번 중의성 단어 각각에 대해서 서로 다른 분류기를 학습 시켜야 한다는 것이 문제가 된다. 딥러닝 모델을 SVM과 마찬가지로 모든 중의성단어별로 각각 구성하고 학습하기에는 상당히 비효율적이다. 6000여개의 딥러닝 모델을 각각 학습하기위해선 상당한 시간이 필요할뿐더러 각 단어별 평균 학습 인스턴스수가 약 200개로 딥러닝 모델을 학습하기에는 충분치 않기 때문이다.

따라서 본 실험에서는 각 중의성 단어별로 딥러닝 모델을 따로 두지 않고 각자 Output Layer만을 따로 가지며 이를 교체해가며 학습하는, Output Layer 교체모델을 시도해보았다(그림4).

이와 같이 딥러닝 모델을 구성할 경우, 단 하나의 모델만으로 모든 중의성단어를 학습 및 처리할 수 있게 된다. 또한 이와 같은 구조는 직관적으로 중의성해소 문제에 대해 “앞뒤 문맥을 고려해야 한다.”, “조사보다는 명사를 좀 더 가중치를 두고 고려해야 한다.” 등과 같은 공통의 풀이법을 학습하는 부분과 “배는 ‘먹다’와 연결될 때와 ‘아프다’와 연결될 때의 의미가 다르다.” 와 같은 개별의 풀이법을 학습하는 부분을 나누어 구성한 것으로 생각할 수 있다.

뿐만 아니라 이 경우 위쪽의 교체부는 각각 자신의 중의성 단어에 대해서만 학습하게 되지만

아래쪽의 고정부는 모든 중의성 단어의 인스턴스에 대해 학습하게 된다. 교체부의 개별 학습 인스턴스 부족 문제를 고정부에서 보완해줄 수 있게 되는 것이다.

본 실험에서는 높은 난이도의 중의성 단어들에 대해서 Output Layer 교체모델을 학습시켜 보고 결과를 SVM 분류기와 비교해보았다. 또한 인스턴스 부족문제를 해결할 수 있을것이라는 직관에 맞게, 학습 인스턴스 수가 적은 중의성 단어들에 대해서만 모델을 학습하고 다시 한번 SVM 분류기와 비교해보았다. 실험에 사용된 딥러닝 모델은 모두 파이썬의 파이토치(Pytorch) 라이브러리를 사용하여 진행하였다.

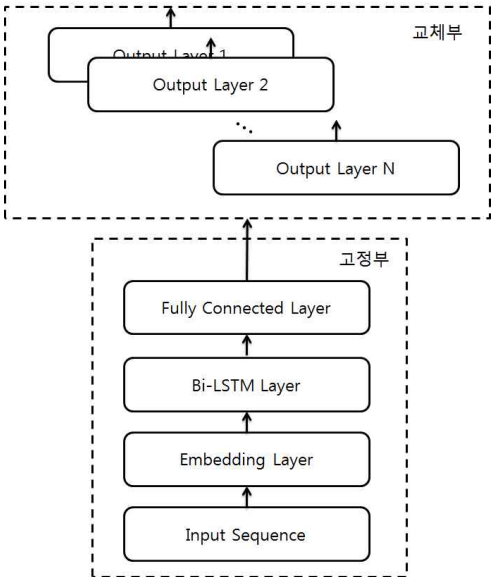


**그림 4** Output Layer 교체모델

## 4. 실험 결과

### 1) 세종말뭉치 통계정보

실험에는 세종말뭉치를 사용하였으며 트레이닝셋과 테스트셋을 9:1로 나누어 사용하였다 (표 3).

| 분류 | 문장 수 | 어절 수 | 형태소 수 | 중의성 단어 수 | Baseline (MFS) |
|---|---|---|---|---|---|
| 트레이닝 셋 | 846,931 개 | 9,945,890 개 | 21,515,393 개 | 23,526 개 | 96% (Accuracy) |
| 테스트 셋 | 94,103 개 | 1,100,559 개 | 2,393,077 개 | 15,219 개 | |

**표 3** 실험 데이터 통계정보

### 2) 평가방법

평가기준은 정확도(Accuracy)를 사용하였으며 계산방식은 식 (2)와 같다.

$$\text{정확도} = \frac{\text{올바르게 분류한 중의성 단어 수}}{\text{평가된 중의성 단어 수}} \qquad (2)$$

### 3) SVM 분류기 실험 결과

세종말뭉치에 나타난 총 23526개의 중의성 단어 중 엔트로피가 0.1 이상인 6109개의 중의성 단어들에 대해 SVM 분류기를 학습하고 자질벡터의 구성방식별 정확도를 측정하였다(표4).

실험결과 Glove word embedding, 윈도우 사이즈 5, 임베딩 차원 200, 병합 방법 concat 이고, Min-Max 벡터를 추가했을 때 정확도가 93.07% 으로 가장 높게 나타났다.

눈여겨볼만한 점은 첫째로 문맥 단어들의 벡터를 병합할 때 element-wise sum 방식보다 concatenation 방식이 압도적으로 뛰어난 성능을 보였다는 점이다. 이는 element-wise sum 방식의 경우 병합과정에서 문맥 단어들의 정보를 상당부분 잃어버리는 반면 concatenation 방식의 경우 잃어버리는 정보 없이 병합되기 때문으로 보인다.

두 번째로 윈도우 사이즈가 무조건 클수록 좋은 것이 아니라는 점 또한 흥미롭다. 같은 조건에서 윈도우사이즈가 2일 때 보다는 5일 때가 뛰어난 성능을 보이지만 윈도우사이즈가 10이 되면 오히려 5일 때보다 성능이 떨어지는 모습을 보였다. 이는 윈도우사이즈가 과도하게 커질 경우 중의성해소에 도움이 되지 않는 문맥들까지 자질벡터에 포함이 되어버리기 때문인 것으로 생각해볼 수 있다.

마지막으로 같은 조건에서 Min-Max 벡터의 추가가 성능향상에 도움이 되는 것을 확인할 수 있었다. 병합과정에서 각 차원별 dominant value는 그 의미가 희석될 수밖에 없는데 Min-Max 값을 따로 뽑아냄으로써 dominant value가 희석되는 것을 보완해주는 것이 Min-Max 벡터의 역할이라고 할 수 있다. 따라서 이처럼 Min-Max 벡터가 있을 때 더 뛰어난 성능을 보였다는 것은 병합과정에서 dominant value가 희석되는 것을 보완해주는 것이 단어 중의성 해소 성능 향상에 도움이 된다는 것을 의미한다고 할 수 있다.

| Min-Max 벡터 | 병합방식 | 윈도우 사이즈 | 임베딩 차원 | 정확도 |
|---|---|---|---|---|
| **O** | **concat** | **5** | **200** | **93.06676964** |
| O | concat | 2 | 200 | 92.9102015 |
| O | concat | 5 | 100 | 92.88913852 |
| X | concat | 5 | 200 | 92.76135646 |
| O | concat | 10 | 200 | 92.74099558 |
| O | concat | 2 | 100 | 92.65955206 |
| O | concat | 10 | 100 | 92.64199958 |
| X | concat | 5 | 100 | 92.52966369 |
| X | concat | 10 | 200 | 92.5212385 |
| X | concat | 2 | 200 | 92.50930282 |

**표 4** 자질벡터 구성방식별 정확도 (상위 10개)

그리고 선행연구와의 비교를 위해 엔트로피가 0.1 미만인 중의성 단어에 대해 MFS 방식을 적용하고 0.1 이상인 중의성 단어에 대해 SVM 분류기를 사용하여 전체 중의성 단어에 대한 정확도를 구해보았다(표5).

그 결과 세종말뭉치 대상 단어 중의성 해소 연구 중 가장 높은 정확도를 보인 선행연구[6]와 같은 수준의 정확도를 보였다. 뿐만 아니라 베이스라인에 비해 정확도가 2.5% 상승 했으며 그 정확도가 98.5%로 매우 높은 편이라고 할 수 있다.

| 엔트로피 | 모델 | 정확도 |
|---|---|---|
| 0.1이상 | SVM | 93.0% |
| **전체** | **SVM+MFS** | **98.5%** |
| 전체 | 선행 연구 (HMM)[6] | 98.5% |
| 전체 | 베이스라인 (MFS) | 96.0% |

**표 5** 전체 엔트로피에 대한 선행연구와의 정확도 비교

### 4) 딥러닝 모델 실험 결과

딥러닝 모델의 경우 전체 중의성 단어에 대해 진행하지 않고 우선 난이도 상위 10~500 개의 중의성 단어에 대해서만 중의성해소를 시도하고 결과를 SVM 분류기와 비교해보았으며 (표7) 사용한 모델의 하이퍼파라미터는 표6과 같다.

실험결과 대상 난이도 상위 단어 10~500개 실험 중 어떤 실험도 SVM의 성능을 뛰어넘지는 못했다. 다만 그 차이는 최소 1.5%에서 최대 4.12%로 크지않은 편이었다.

이후 딥러닝 모델이 적은 학습인스턴스를 가지는 중의성단어들에 대해 얼만큼의 성능을 보이는지 알아보기 위해 난이도 상위 500개의 단어 중 학습인스턴스 수가 M개보다 적은 단어들만 모아 학습하고 평가해보았다(표8).

그 결과 학습 인스턴스 수가 적은 단어들만 모아서 학습 및 평가를 진행하여도 딥러닝 모델이 SVM 분류기의 성능에는 미치지 못하는 것으로 나타났다. 결과적으로 실험 전 예상했던 바와 달리 높은 난이도, 적은 인스턴스를 가지는 중의성 단어의 경우에도 SVM 분류기의 성능이 딥러닝 모델보다 뛰어난 것으로 볼 수 있다.

| 하이퍼파라미터 | 사용 값 |
|---|---|
| 윈도우 사이즈 | 5 |
| 워드 임베딩 방식 | Glove |
| 워드 임베딩 차원 | 200 |
| Bi-LSTM 히든 레이어 차원 | 2*128 |
| Fully Connected Layer 차원 | 64 |
| 출력 차원 | 가변 (타겟 단어의 의미 수) |
| 로스 펑션 | Cross Entropy |
| Optimization 알고리즘 | Adam (learning rate 0.001) |
| Epochs | Early Stopping (patience : 10) |

**표 6** 딥러닝 모델의 하이퍼파라미터

| 모델 \ N | 10 | 20 | 100 | 200 | 500 |
|---|---|---|---|---|---|
| 딥러닝 | 75.78 | 77.42 | 80.16 | 79.77 | 83.22 |
| SVM | 77.60 | 79.40 | 83.37 | 83.89 | 84.72 |
| 차이 | -1.82 | -1.98 | -3.21 | -4.12 | -1.5 |

**표 7** 난이도 상위 N 단어에 대한 중의성 해소 정확도 비교 (단위: %)

| 모델 \ M | 20 | 30 | 50 |
|---|---|---|---|
| 딥러닝 | 47.23 | 48.92 | 56.70 |
| SVM | 55.27 | 58.57 | 61.68 |
| 차이 | -8.04 | -9.65 | -4.98 |

**표 8** 난이도 상위 500단어 중 학습 인스턴스 수가 M개 미만인 단어들에 대한 정확도 비교(단위: %)

## 5. 결론

본 연구에서는 SVM 분류기와 딥러닝모델을 이용하여 세종말뭉치 내의 중의성 단어들에 대해 단어 중의성 해소를 시도하고 그 성능을 분석해보았다.

SVM 분류기의 성능은 기대이상이었다. 엔트로피 0.1이상 단어들에 대해 약 93%의 정확도를 보였으며 이는 MFS와 함께 사용시 전체 단어들에 대해 98.5%의 정확도를 보이는 수치이다. 선행연구 중 가장 뛰어난 정확도를 보인 연구가 98.5%임을 감안할 때, 아주 뛰어난 결과라고 할 수 있다. 특히 최상의 자질벡터 구성을 찾는 과정에서 윈도우 사이즈가 무조건 클수록 좋은 것이 아니라는 사실과, 병합시 concatenation 방식이 더 좋은 성능을 보이며 Min-Max 벡터를 추가할 경우 성능이 더 좋아진다는 사실 등을 확인할 수 있었다. 다만 애초에 베이스라인 정확도가 96%에 이를 만큼 세종말뭉치 자체의 중의성해소 난이도가 높은편이 아니라는 사실은 한계점이라 할 수 있다.

반면 딥러닝 모델의 성능은 기대이하였다. 모든 중의성단어에 대해서 성능을 측정할 수는 없었지만 난이도 상위 500개의 단어에 대해 학습 및 평가해본 결과 모두 SVM에 비해 뒤처지는 성능을 보이고 말았다. 특히 가장 기대했던 부분인 “적은 학습 인스턴스를 가진 단어들을 함께 학습할 때 전체적인 풀이 경향을 학습할 수 있을 것이다” 라는 가설을 입증할 수 있을만큼 만족스런 결과를 보이지 못했다. 이는 단어 중의성해소 문제에서 전체의 풀이경향을 학습하는 것 만큼 개별 단어의 고유한 풀이법을 학습하는 것 또한 중요하다는 사실을 보여주는 결과라고 할 수 있다.

다만 애초에 데이터가 딥러닝 모델을 충분히 학습시킬 수 있을 만큼 많지 않았다는 점을 감안하면 딥러닝 모델의 가능성을 확인할 수 있는 실험이었다고 할 수 있다. 세종말뭉치에 등장하는 중의성단어별 평균 학습인스턴스 수는 약 200개이다. 일반적으로 딥러닝을 활용하는 분류문제의 학습인스턴스 수가 수 만개가 넘어가는 것을 생각하면 200개는 턱없이 부족한 수이다. 그럼에도 불구하고 SVM 분류기에 비해 정확도가 크게 뒤지지 않는 성능을 보인 것은 본 실험의 딥러닝 모델이 데이터 부족문제를 일부 보완할수 있을만한 가능성을 보여준것이라 할 수 있다.

본 연구에서 보인 SVM 분류기의 뛰어난 성능과 딥러닝 모델의 가능성을 활용한다면 앞으로 한국어 단어 중의성 해소 문제에서 더욱 다양한 시도가 가능할것으로 기대된다. 다만 현재 연구를 위한 한국어 말뭉치 여건이 열악한 만큼 양질의, 방대한량의 새로운 말뭉치가 계속해서 제작되는 것이 우선 과제이며 양질의 말뭉치가 바탕이 된다면 다양한 시도를 통해 자연어처리 전반에 걸쳐서 큰 발전을 이룰 수 있을 것으로 기대된다.

# 참고문헌


[1] 허정, & 옥철영. (2001). 사전의 뜻풀이말에서 추출한 의미정보에 기반한 동형이의어 중의성 해결 시스템. 정보과학회논문지: 소프트웨어 및 응용, 28(9), 688-698.

[2] 김준수, & 옥철영. (2005). 인공지능: 정제된 의미정보와 시소러스를 이용한 동형이의어 분별 시스템. 정보처리학회논문지 B, 12(7), 829-840.

[3] 배영준, 최호섭, 송유화, & 옥철영. (2011). 사전 뜻풀이를 이용한 용언 의미 군집화. 인지과학, 22(3), 271-298.

[4] 강상욱, 김민호, 권혁철, 전성규, & 오주현. (2015). 세종 전자사전과 한국어 어휘의미망을 이용한 용언의 어의 중의성 해소. 정보과학회 컴퓨팅의 실제 논문지, 21(7), 500-505.

[5] 신준철, & 옥철영. (2016). 한국어 어휘의미망 (UWordMap) 을 이용한 동형이의어 분별 개선. 정보과학회논문지, 43(1), 71-79.

[6] 신준철, (2014), 기분석 부분어절 사전 기반의 형태소 분석 및 음절-형태소 전이 확률 기반 품사·동형이의어 태깅, 울산대학교 대학원.

[7] 박준혁, & 이성욱. (2016). 지지벡터기계를 이용한 단어 의미 분류. 정보처리학회논문지. 소프트웨어 및 데이터 공학, 5(11), 563-568.

[8] 강명윤, 김보겸, & 이재성. (2015). Word2Vec를 이용한 단어 의미 모호성 해소. 한국정보과학회 언어공학연구회:학술대회논문집(한글 및 한국어 정보처리), 27, 81-84,

[9] 이현아. (2014). 가변 크기 문맥과 거리가중치를 이용한 동형이의어 중의성 해소. 한국마린엔지니어링학회지, 38(4), 444-450.

[10] Joachims, T. (1998, April). Text categorization with support vector machines: Learning with many relevant features. In European conference on machine learning (pp. 137-142). Springer, Berlin, Heidelberg.