Research Paper https://doi.org/10.23287/KJDH.2024.1.1.6

Kiwi: 통계적 언어 모델과 Skip-Bigram을 이용한 한국어 형태소 분석기 구현

Kiwi: Developing a Korean Morphological Analyzer Based on Statistical Language Models and Skip-Bigram

이민철(Lee, Min-chul)*
0009-0009-2898-5706

목차

초록

한국어 형태소 분석 시 모델이 마주하는 어려움 중 하나는 모호성이다. 이는 한국어에서 기저형이 전혀 다른 형태소 조합이 동일한 표면형을 가질 수 있기 때문에 발생하며 이를 바르게 분석하기 위해서는 문맥을 고려하는 능력이 모델에게 필수적이다. 형태소 분석기 Kiwi는 이를 해결하기 위해 근거리 맥락을 고려하는 통계적 언어 모델과 원거리 맥락을 고려하는 Skip-Bigram 모델을 조합하는 방식을 제안한다. 제안된 방식은 모호성 해소에서 평균 정확도 86.7%를 달성하여 평균 50~70%에 머무르는 기존의 오픈소스 형태소 분석기, 특히 딥러닝 기반의 분석기들보다도 앞서는 결과를 보였다. 또한 최적화된 경량 모델을 사용한 덕분에 타 분석기보다 빠른 속도를 보여 대량의 텍스트를 분석하는 데에도 유용하게 쓰일 수 있다. 오픈소스로 공개된 Kiwi는 전술한 특징들 덕분에 텍스트 마이닝, 자연어처리, 인문학 등 다양한 분야에서 널리 사용되고 있다. 본 연구는 형태소 분석의 정확도와 효율성을 모두 개선했으나, 미등록어 처리와 한국어 방언 분석 등의 과제에서 한계를 보여 이에 대한 추가 보완이 필요하다.

주제어: 한국어, 자연어처리, 형태소 분석기, 모호성 해소, 언어 모델

* 카카오 Language Model Application 팀, 연구원, phil.os@kakaocorp.com

Abstract

One of the challenges faced by models in Korean morphological analysis is ambiguity. This arises because different combinations of morphemes with completely different base forms can share the same surface form in Korean, necessitating the model's ability to consider context for accurate analysis. The morphological analyzer Kiwi addresses this issue by proposing a combination of a statistical language model that considers local context and a Skip-Bigram model that considers global context. This proposed method achieved an average accuracy of 86.7% in resolving ambiguities, outperforming existing open-source morphological analyzers, particularly deep learning-based ones, which typically achieve between 50-70%. Additionally, thanks to the optimized lightweight model, Kiwi shows faster speeds compared to other analyzers, making it useful for analyzing large volumes of text. Kiwi, released as open source, is widely used in various fields such as text mining, natural language processing, and the humanities due to these features. Although this study improved both the accuracy and efficiency of morphological analysis, it shows limitations in handling out-of-vocabulary problem and analyzing Korean dialects, necessitating further improvements in these areas.

Keyword: Korean, NLP, Morphological Analyzer, Disambiguation, Language Model

1. 서론

1.1 형태소 분석기와 형태소

형태소 분석은 자연어처리(Natural Language Processing, NLP) 분야에서 가장 기본적이면서도 중요한 작업이다. 형태소 분석은 주어진 텍스트를 형태소 단위로 분할하고 각 형태소에 대해 품사를 부착하는 작업으로 정보 검색, 문서 분류, 감성 분석, 기계 번역, 음성 인식, 대화 시스템 등 다양한 NLP 작업에 필수적으로 사용된다. 특히 한국어의 경우 체언에는 조사가 붙고 용언에는 어미가 붙어 그 형태가 다양하게 변형되는 특징이 있어 형태소 분석을 통해 그 원형을 복원하는 작업이 필수적이다. 이 때문에 한국어 NLP 연구가 본격적으로 시작된 1980년대 이후 다양한 연구자들이 형태소 분석기 개발에 많은 노력을 기울여왔다.

이렇게 연구된 다양한 분석기들은 모두 '형태소 분석기'라는 이름으로 불리고 있으나 각 연구마다 목표로 하는 분석 단위가 상이하며 실제로는 목표 분석 단위가 형태소가 아닌 경우도 있다. 언어학적으로 형태소는 의미를 지닌 가장 작은 말의 단위로 정의된다. 예를 들어 '감기약'이라는 단어는 '감기'라는 형태소와 '약'이라는 형태소로 구성된다. 따라서 엄밀하게는 '감기약'이라는 단어를 '감기'와 '약'으로 단어를 분해해주는 분석기만을 형태소 분석기라고 불러야 하겠지만, 조사만 분리하여 단어까지만 추출해주는 분석기들까지도 관습적으로 형태소 분석기라고 불리고 있다. 이는 초창기 한국어 형태소 분석기 연구가 주로 정보 검색을 위한 색인이나 키워드 추출 등의 목적으로 시도되었기에 체언에서 조사를, 용언에서 어미를 분리하여 기본형을 찾는 것에 더 집중했던 점에서 기인한다. 세종 품사 태그 집합[1]의 가이드라인에서도 표준국어대사전에 명사로 등재된 표제어라면 형태 분석 단위인 일반명사(NNG)가 될 수 있도록 정의하여 단어 내부의 형태소 분석까지는 고려하지 않고 있다. 이에 세종 품사 태그 집합에 기반한 형태소 분석기들은 형태소와 단어를 절충하여 분석 단위로 설정하고 있으며 본 논문에서도 이런 맥락에서 형태소와 사전 표제어를 분석 단위로 하는 분석기에 형태소 분석기라는 표현을 사용한다.

1.2 한국어 형태소 분석의 중요성과 어려움

한국어는 교착어로 하나의 형태소가 그 자체로 단어를 이루기도 하고 여러 개의 형태소가 결합해 한 단어를 이루기도 하며, 특히 여러 형태소가 결합할 때 축약하여 그 표면형이 크게 달라지기도 한다. 이런 복잡성이 가장 두드러지는 것은 용언의 활용이다. 예를 들어 동사 '불다'는 그 어간이 '불-'이지만 실제 문장에서는 '불었다', '불까요', '부셨습니다', '불던', '불' 등 다양한 형태로 등장한다. 따라서 다양한 형태에서 실제 의미를 담고 있는 '불-'과 문법적인 기능을 수행하는 어미들을 분리하여 추출하기 위해서는 형태소 분석이 필수적이다.

형태소 분석은 NLP에서 가장 기본적인 작업이지만 한국어의 경우 이를 수행하는 것이 다른 언어에 비해 어렵다. 단어의 경계가 띄어쓰기를 통해 명백하게 구분되는 영어와는 달리 한국어에서는 띄어쓰기의 경계가 단어와 불일치한다. 때문에 맞춤법 상에 명확한 띄어쓰기 규범이 규정되어 있음에도 언중들이 이를 헷갈려 하기에 사람에 따라 다양한 띄어쓰기 오류가 나타난다. 또한 용언의 활용 과정에서 서로 다른 형태소 조합이 동일한 표면형을 가지는 경우도 많아 모호성 해소가 동반되어야만 올바른 형태소를 도출할 수 있는 경우도 많다. 예를 들어 '풍선을 부셨습니다'에서 '부셨습니다'는 '불-'에 '-시-', '-었-', '-습니다'가 결합된 것이지만 '눈이 부셨습니다'에서 '부셨습니다'는 '부시-'에 '-었-', '-습니다'가 결합된 형태이다. 이 둘은 전혀 다른 동사의 활용형이지만 표면형은 동일하기 때문에 문맥을 통해 의미를 파악해야 한다. 이처럼 한국어 형태소 분석은 단순히 형태소를 분리하는 것을 넘어서 문맥을 고려하여 형태소를 구분하고 품사를 부착해야 하는 어려운 작업인 것이다.

최근 BERT, GPT를 비롯한 다양한 사전학습 언어 모델(Pretrained Language Model, PLM)이 NLP의 핵심 모델이 되면서 형태소를 고려하지 않는 토큰화(Tokenization) 기법[2]만을 사용해 다양한 NLP 과제를 해결하는 것이 대세가 되고 있다. 특히 모델의 크기가 커질수록 토큰화 기법에 관계 없이 모든 NLP 과제에서 높은 성능을 보이는 것[3]이 확인되었기에 토큰화 기법보다는 모델의 크기와 학습 데이터의 양이 더 중요한 요소로 인식되는 경향이 있다. 그러나 이런 PLM에서도 모델의 크기가 동일한 상황이라면 형태소를 고려한 모델이 그렇지 않은 모델보다 언어 이해도가 더 높다는 것이 한국어[4]와 영어[5] 등 다양한 언어에 대해서 입증되고 있다. 또한 형태소에 기반한 접근법은 언어 모델이 언어를 이해하는 과정을 설명하기 용이하다는 특징도 있다. 거대언어모델에 대한 연구가 크기에서 효율성으로, 또 블랙박스에서 설명가능한 모델로 이동하고 있는 것을 고려할 때 한국어 형태소 분석의 중요성은 여전히 높다고 할 수 있다.

본 논문에서는 한국어 모호성 해소에 높은 성능을 개선하기 위해 맥락을 고려할 수 있는 언어 모델을 제안한다. 구체적으로 근거리 맥락을 고려하는 통계적 n-gram 언어 모델과 원거리 맥락을 고려하는 Skip-Bigram 모델을 조합하는 접근법을 통해 형태소를 분석하는 방법을 설명하며 이를 기반으로 구현된 오픈소스 한국어 형태소 분석기 **Kiwi**를 소개한다. 이어서 다양한 분야에서 **Kiwi**가 어떻게 활용되고 있는지에 대해 보이고 마지막으로 제안된 방법의 한계점과 향후 개선 방향에 대해 논의하며 이 글을 마무리한다.

2. 관련 연구

2.1 형태소 분석 방법론

한국어 형태소 분석기는 크게 규칙 기반과 통계 기반, 그리고 딥러닝 기반 기법으로 나누어 볼

수 있다. 규칙 기반 기법에서는 연구자가 수작업으로 선별한 규칙에 기반해 형태소를 분석한다. 규칙에 일치하는 입력에 대해서는 높은 정확도를 달성할 수 있지만 연구자가 넓은 범위의 규칙을 작성해야 하므로 작업량이 많고 새로운 형태소나 신조어 등에 대한 대응이 어렵다는 단점이 있다. 이에 통계 기반 방법이 제안되었는데, 이 방법은 대량의 말뭉치를 이용해 형태소의 분포를 학습하고 이를 바탕으로 형태소 분석을 수행한다.

통계 기반 방법은 연구자가 모든 규칙을 작성하지 않아도 충분히 높은 정확도를 달성할 수 있다는 장점이 있으나 형태 분석이 수행된, 충분한 규모의 말뭉치가 필수적이다. 다행히 21세기 세종 계획으로 대량의 형태 분석 말뭉치가 확보된 덕분에 다양한 통계 기반 접근법이 연구될 수 있었다. 특히 최근에는 이 말뭉치들을 이용해 여러 층의 신경망(Neural Network)으로 구성된 딥러닝 모델을 학습시키는 접근법도 널리 연구되고 있다. 각 기법별 대표적 연구를 정리하자면 다음과 같다.

2.1.1 규칙 기반 분석

일반적으로 형태소 분석기는 크게 입력 텍스트로부터 형태소의 가능한 조합을 분리해내고(형태소 분리), 그 중 가장 적절한 형태소 배열을 선택(품사 부착)하는 과정을 통해 형태소를 분석해낸다. 규칙 기반 분석에서는 대체로 이 두 과정 모두를 연구자가 설계한 규칙에 의해 수행한다. 대표적으로 김성용[6]과 권오욱[7]은 Tabular Parsing에 기반해 형태소 분리를 수행하고 다양한 제약조건을 바탕으로 최적의 후보를 선택하는 방식을 사용했다. 효율적인 형태소 분리를 위해 양방향 최장일치법[8]이나 부분 어절의 기분석[9] 방법이 제안되기도 했다. 또한 형태소 분리 단계에서의 과생성을 해결하기 위해 언어학적 정보를 사용하는[10] 기법과 형태소 간의 계층 구조[11]를 이용한 연구도 있었다. 또한 알려지지 않은 형태소 문제를 해결하기 위해 음절 패턴 기반의 예측 모델[12]이 제안되기도 했다.

2.1.2 통계 기반 분석

통계 기반의 접근법은 말뭉치로부터 모델을 학습할 수 있으므로 연구자가 일일이 규칙을 작성할 필요가 없고 특정 태그집합에 종속적이지 않기 때문에 연구의 확장성이 높다. 때문에 다양한 한국어 말뭉치(ETRI, KAIST, 21세기 세종계획)들이 구축되면서 형태소 분석기 연구의 관심은 통계 기반으로 옮겨 갔다. 이도길과 임해창[13]은 표면형으로부터 형태소를 생성하는 확률 모델을 세우고 이를 최대우도법으로 학습시킴으로써 한국어 형태소 분석이 연구자가 별도로 작성한 규칙 없이 확률 모델만으로도 수행될 수 있음을 보였다. 이재성[14]은 형태소 분리 과정에서 원형 복원을 별도 단계로 나누어 각 단계를 단순화시키고 형태소 전이 확률을 기반으로 형태소 분리함으로써 분석 성능을 높였다.

한편 형태소 분리에는 어휘집(Lexicon)을 사용하고 품사 부착에는 통계적 모델을 사용한 하이브리드 접근법도 제안되었다. 대표적인 것이 Kudo의 연구[15]로 그는 규칙에 기반한 어휘집을 통해 텍스트를 형태소 격자로 구성하고, 조건부 무작위장(Conditional Random Field, CRF)를 이용해 품사를 부착하는 방식을 사용했다. 그는 이 기법으로 일본어 형태소 분석기인 MeCab[16]을 개발하여 오픈소스로 공개했다. 특히 MeCab은 확장성이 높아 어휘집을 교체하고 CRF 모델을 새로 학

습하면 다른 언어를 분석하는 데에도 사용이 가능하다. 덕분에 국내에서도 MeCab용 한국어 형태소 사전을 구축하려는 시도인 은전한닢 프로젝트[17]가 진행되었다. 이 프로젝트의 결과물로 2013년 공개된 한국어 오픈소스 형태소 분석기 MeCab-ko는 빠른 속도와 높은 정확도로 현재도 널리 쓰이고 있다.

나승훈[18]은 Kudo와 유사하게 격자를 사용하지만 품사 부착에 Averaged Perceptron을 사용한 뒤 미등록어 분리를 위한 후처리를 추가한 한국어 형태소 분석기를 제안했고 이를 통해 높은 정확도를 달성할 수 있었다. 한편 심광섭[19]은 음절 단위에 CRF를 적용하여 품사를 부착하는 방식을 제안했다. 이 방식은 음절에 바로 CRF를 적용하므로 별도의 형태소 분리 단계가 필요 없어 높은 성능을 유지하면서도 시스템 구조를 단순화할 수 있다는 장점이 있다.

2.1.3 딥러닝 기반 분석

최근 심층신경망(Deep Neural Network)이 NLP 과제에 있어서 매우 뛰어난 성능을 보인다는 것이 확인됨에 따라 형태소 분석에도 신경망 모델을 도입하려는 다양한 시도가 이어지고 있다. 딥러닝 기반 분석기에서 두드러지는 접근법은 end-to-end로 이는 신경망 외의 다른 중간 과정을 거치지 않고 입력 텍스트로부터 바로 형태소 분석 결과를 출력해내는 방법들을 일컫는 표현이다. 대표적으로 서대룡[20]은 심광섭이 시도한 음절 기반 접근법에 신경망을 적용하였다. 순환신경망(Recurrent Neural Network, RNN) 모델의 일종인 BiLSTM에 CRF를 결합시켰고 한글 자모를 분리하여 음절단위로 입력함으로써 자모가 바뀌는 오타에도 강건한 모델을 구축할 수 있었다. 카카오에서 공개한 오픈소스 형태소 분석기인 Khaiii[21] 역시 유사한 아이디어를 바탕으로 한다. 대신 Khaiii는 RNN 대신 병렬처리가 가능한 합성곱 신경망(Convolutional Neural Network, CNN) 모델을 채용하여 속도를 높여 실용적인 시스템을 구축했다. 또한 훈련 데이터셋인 세종 계획 말뭉치의 오류를 수정하여[22] 모델의 정확도를 크게 개선했고 쉽게 사용할 수 있도록 Python 바인딩을 제공한 덕분에 Khaiii는 현재도 다양한 곳에서 널리 사용되고 있다.

한편 기계 번역에 주로 사용되는 구조인 Sequence-to-sequence를 채용한 연구도 있다. 이건일[23]은 형태소 분석을 입력 텍스트를 품사가 부착된 출력 텍스트로 변환하는 번역 과제로 보고 Sequence-to-sequence 모델을 학습하였다. 정확도는 기존의 판별 모델 기반의 분석기를 앞서지는 못하였지만 형태소 분리와 품사 부착 등 여러 단계로 구성된 기존의 형태소 분석기와는 달리 어휘집이나 규칙 없이 모델만으로 입력 텍스트에서 바로 품사 부착된 결과를 생성하는 것이 충분히 가능하다는 것을 보였다는 점에서 이 연구는 의미가 있다. 최병서[24]는 이 연구에서 더 나아가 입력 자질로 음절뿐만 아니라 자모와 음절 Bigram을 함께 사용함으로서 띄어쓰기 오류나 신조어가 포함된 문장의 분석 성능을 높였다.

2019년 BERT[25]가 발표된 이후 BERT를 한국어 형태소 분석에 사용하려는 연구도 등장했다. 민진우[26]와 박천음[27]의 연구가 대표적이다. 두 연구는 모두 BERT에 LSTM-CRF를 결합하여 사용한다는 점에서는 동일하지만 민진우의 연구는 입력 단위를 Subword로, 박천음의 연구는 입력 단위를 음절로 설정했다는 차이가 있다. 특히 후자는 입력 단위가 음절이라는 점에서 이전에 심광섭이나 서대룡이 제안한 모델 구조와 유사하지만 핵심 모델이 BERT로 교체됨에 따라 정확도가 크게 향상되었다.

2.2 오픈소스 한국어 형태소 분석기

형태소 분석기는 검색 엔진이나 자동 색인과 같은 정보 검색 시스템에 필수적으로 사용되기 때문에 1990년대까지는 웹 검색 서비스를 제공하는 포털 사이트 업체나 기업 맞춤형 검색 엔진을 구축하는 업체에서 독자적으로 개발하여 사용했다. 이때까지만 해도 형태소 분석기는 상용 소프트웨어로만 제공되었기 때문에 연구자들이나 개발자들이 자유롭게 사용하거나 개선하기가 어려웠다. 그러나 1999년 한국어 형태소분석기 평가대회인 MATEC99[28]가 개최되고, 대회에 제출되었던 형태소 분석기 중 하나인 KAIST SWRC 연구실의 Hannanum[29][30]이 공개되면서 오픈소스 한국어 형태소 분석기의 포문을 열었다. 그 이후로 서울대 IDS 연구실에서 개발한 꼬꼬마[31][32] 형태소 분석기가 공개되었다. 오픈소스 형태소 분석기는 누구나 자유롭게 사용할 수 있고 필요에 따라 기능을 추가하거나 다른 시스템과 손쉽게 연동할 수 있다는 장점이 있어 많은 연구자와 개발자들에게 환영을 받았다.

학계뿐만 아니라 오픈소스 커뮤니티에서도 다양한 형태소 분석기를 개발하여 공개했다. 대표적인 것으로는 ElasticSearch나 Solr 등의 검색 엔진과의 연동을 목적으로 개발된 Arirang[33], Daon[34], 통계 분석에서 주로 쓰이는 R언어와 연동할 목적으로 개발된 RHINO[35], 범용적인 목적으로 개발된 Komoran[36]과 은전한닢 프로젝트를 통해 구축된, MeCab의 한국화 버전인 MeCab-ko[37], MeCab-ko를 Scalar로 포팅한 seunjeon[38], Mecab-ko를 Python으로 재구현한 PeCab[39], 그리고 본 논문에서 소개하는 **Kiwi**[40] 등이 있다.

기업에서 직접 개발하여 공개한 분석기도 있다. Twitter에서 개발한 twitter-korean-text가 대표적인 사례로 공개 후 관리주체가 오픈소스 커뮤니티로 넘어가며 이름이 open-korean-text[41]로 변경되었다. 또한 카카오에서 개발한 Khaiii[42]도 있다. 이는 오픈소스 형태소 분석기 중 최초로 딥러닝 모델을 채용한 사례로, CNN 모델을 사용하여 빠른 속도와 높은 정확도를 동시에 달성한 것이 특징이다. 또한 바이칼에이아이와 한국언론진흥재단이 공동으로 개발한 상용 형태소분석기 바른(Bareun)[43]도 주목할 만하다. 바른은 오픈소스로 공개되지는 않았지만 비상업적 용도로는 무료로 배포하고 있으며 BERT에서 사용되는 트랜스포머 아키텍처를 기반 모델로 채택하여 정확도를 높이 끌어올린 것으로 유명하다.

3. Kiwi 소개

이 장에서는 **Kiwi** 형태소 분석기의 주요 특징과 그 구조에 대해서 소개한다. **Kiwi**(Korean Intelligent Word Identifier)는 누구나 쉽게 사용할 수 있는 범용적인 한국어 분석기를 목표로 하는 오픈소스 형태소 분석기이다. 구조적으로 **Kiwi**는 Kudo 및 나승훈 제안한 형태소 분석기와 유사하게 형태소 분리와 품사 부착 두 단계를 거쳐서 분석을 수행하지만, 품사 부착에 CRF나 Averaged Perceptron과 같은 판별 모델로 형태소 간 전이 점수를 계산하는 대신, 언어 모델을 직접 사용하여 형태소의 가능한 조합을 확률적으로 평가한다는 점에서 차이가 있다. 이론적으로는 주어진 이전 형태소 배열에서 다음 형태소를 확률적으로 예측할 수 있는 언어 모델이라면 어떤 것이든 사용할 수 있지만, 범용적인 환경에서 널리 사용될 수 있도록 대규모 연산이 필요하지 않은 경량 Modified Kneser-Ney n-gram 언어 모델[44] 및 그 변형을 채택하였다. 형태소 분리 단계에서는 이전 연구들과 마찬가지로 어휘집과 형태소 결합 규칙을 사용했다.

최근 널리 연구되고 있는 딥러닝 기반의 end-to-end 접근법이 일반적인 문장뿐만 아니라 미등록어나 신조어가 포함된 문장에 대해서도 형태소를 비교적 정확하게 추출해내고 있음에도 Kiwi에서 이를 채택하지 않은 것에는 크게 세 가지 이유가 있다.

첫째 end-to-end 방식은 형태소의 재현율(Recall)을 높일 순 있지만 정확률(Precision)은 오히려 떨어뜨릴 수 있다. end-to-end 방식이 미등록어나 신조어를 분석해낼 수 있다는 것은 어휘집에 구애받지 않고 분석을 실시한다는 것이고 이는 반대로 어휘집에 있는 형태소를 추출해야 하는 상황에서도 어휘집에 없는 형태소를 추출해낼 가능성이 있다는 것이다.

둘째 딥러닝 기반 방식은 예측 가능성과 통제 가능성이 낮다. 첫 번째 이유와 연결되는 것으로 어휘집에 없는 형태소를 생성하는 것이 가능하다 보니 특정 입력에 대해 사용자가 기대하는 형태소가 나올 것이라는 보장이 없고 이를 통제하기 어렵다. 반대로 어휘집이나 규칙에 기반한 방법은 항상 어휘집 혹은 규칙에 기재된 형태소만이 출력의 대상이 되므로 결과가 예측 가능하며 상황에 따라 어휘집 및 규칙을 수정하는 것으로 쉽게 결과를 통제할 수 있다.

마지막으로 딥러닝 기반 방식은 모델 크기 및 연산량이 기존 방식 대비 크다. 이는 모델을 학습하고 사용하는 데에 많은 컴퓨팅 자원이 필요하다는 것을 의미하며, 이는 연구자나 개발자가 쉽게 사용하거나 개선하기 어렵다는 것으로 이어진다. 따라서 결과의 예측 가능성과 통제 가능성을 높이고 연산량을 줄이기 위해 Kiwi는 어휘집과 규칙을 통한 형태소 분리를 실시하고, 그리고 통계적 경량 언어 모델로 품사 부착을 수행하는 방식을 채택했다.

다만 end-to-end를 채택하지 않은 만큼 미등록어나 신조어를 정확하게 분석하기 위해서는 사용자가 직접 어휘집에 이를 등록하는 작업이 필요하다. 덕분에 사용자가 분석 결과에 대한 통제권을 가질 수는 있지만 어휘집 관리 작업이 번거롭기 때문에 Kiwi에서는 편의성을 위해 분석 대상 말뭉치 내에서 자주 등장하는 미등록어 문자열 패턴을 찾아 어휘집에 추가할 후보를 제안하는 기능을 제공하고 있다.

3.1 Kiwi의 주요 특징

Kiwi는 범용적인 환경에서 널리 사용할 수 있는 한국어 형태소 분석기를 목표로 하며 다음과 같은 특징을 지닌다.

- 다양한 환경에서 사용 가능: Windows, Linux, macOS 등 주요 운영체제를 모두 지원하며 설치에 필요한 별도의 의존 관계가 없다. 또한 C, C++, C#, Python, Java, Go, R 등 다양한 언어에서 사용 가능하도록 API를 제공한다.
- 빠른 분석 속도: 핵심 모듈이 C++로 구현되어 있고 여러 가지 최적화가 적용되어 있어 빠른 분석 속도를 제공한다.
- 높은 모호성 해소 정확도: 언어 모델을 채택하였기 때문에 맥락을 고려할 수 있어 문맥에 따라 형태소 분석 결과가 달라져야 하는 경우에도 현재 공개된 형태소 분석기들 중에서 가장 높은 정확도를 보인다. 이에 대해서는 4.2에서 자세히 다룬다.
- 독립적인 모듈 구조: 형태소 분리와 품사 부착이 독립적인 모듈로 구성되어 있어서 각 모듈을 교체하는 것이 가능하다. 예를 들어 품사 부착에 사용되는 언어 모델은 전체 시스템에서 차지하는 메모리 및 연산량이 큰데, 이를 작은 크기의 모델로 교체하여 전체 시스템을 경량 형태소 분석기로 사용하는 것 등이 가능하다.

실제로 현재 공개되는 Kiwi 배포판에는 KNLM과 SBG라는 두 종류의 언어 모델을 탑재하여 필요에 따라 골라 사용하는 것이 가능하도록 하였다. KNLM은 모호성 해소 성능은 다소 낮지만 속도가 빠르고 반대로 SBG는 모호성 해소 성능은 높지만 속도는 느리다.

- 오타 강건성: 언어 모델 및 규칙에 기반하여 오타 교정을 수행할 수 있어 오타가 포함된 문장에 대해서도 높은 정확도를 보인다.
- 띄어쓰기 오류 교정: 띄어쓰기가 잘못된 문장을 교정하는 것이 가능하기에 띄어쓰기 오류가 포함된 문장에 대해서도 높은 정확도를 보인다.
- 형태소 결합 기능 제공: 일반적인 형태소 분석기와는 다르게 텍스트를 형태소 단위로 분해하는 기능뿐만 아니라 분해된 형태소를 결합하여 텍스트를 복원하는 기능까지 함께 제공하여 한국어 텍스트를 형태소 단위로 수정할 경우 유용하게 쓰일 수 있다.

이 밖에도 효율적인 탐색을 위해 내부적으로 한글을 초성, 중성은 결합하고 종성만 분리하여 표현하는 방법을 채택하였고, 품사 태그 집합은 세종계획의 품사 태그 집합을 기반으로 하되 유용한 품사 태그를 추가하여 확장하였다. 이에 대해서는 아래에서 자세히 설명한다.

3.1.1 Kiwi의 한글 표현 방법

한글은 자음과 모음이 개별적으로 존재하지만 이를 한 음절로 모아서 표기하는 독특한 특징이 있다. 이 때문에 컴퓨터 내에서 한글을 인코딩하는 방법에 대해서도 개별 자모를 최소 단위로 보는 방법(조합형)과 음절을 최소 단위로 보는 방법(완성형)이 각각 제안되어 어느 쪽이 더 효율적인지 경쟁한 바가 있다. 유사한 논쟁이 형태소 분석기 내에서도 존재한다. 분할 단계에서 사용하는 형태소 사전의 문자 집합을 설정하는 문제가 바로 그것이다. 분할 단계에서는 텍스트 내에 잠재하는 형태소 후보를 최대한 추출하기 위해 대규모의 형태소 사전을 이용한다. 이 과정을 효율적으로 수행하기 위해 형태소 사전은 트라이(Trie)라는 자료구조로 저장된다. 트라이 구조의 탐색 속도는 문자 집합의 크기가 작을수록 또 입력 문자열의 길이가 짧을수록 더 빨라진다.

그런데 문자 집합을 어떻게 설정하느냐에 따라 입력 문자열의 길이가 함께 달라지기 때문에 최적의 문자 집합을 설정하는 것은 단순하지 않은 문제가 된다. 예를 들어 음절 단위를 문자 집합으로 잡으면 '글자'라는 텍스트는 문자 '글'과 '자'로 이뤄진 길이 2의 문자열이 되겠지만, 자소 단위를 문자 집합으로 잡으면 'ㄱ', 'ㅡ', 'ㄹ', 'ㅈ', 'ㅏ'로 이뤄진 길이 5의 문자열이 된다. 음절 단위를 문자 집합으로 잡아서 문자 집합을 크게 가져가는 대신 입력 문자열의 길이를 짧게 하는 것과 자소 단위를 문자 집합으로 문자 집합을 작게 하는 대신 입력 문자열의 길이를 길게 하는 것 중 더 효율적인 것을 찾아 선택해야 하는 것이다.

표 1은 다양한 한글 표현 방법을 비교한 것으로, Kiwi는 여기서 음절과 자소의 절충안인 '초성과 중성은 결합하되 종성만 분리하여 표현하는 방법'을 선택했다. 효율적인 메모리 활용을 위해서는 문자 집합의 크기를 작게 유지하는 것이 필수인데, 자소의 경우 문자 집합은 67개(초성 19개 + 중성 21개 + 종성 27개)로 충분히 작지만 문자열 길이가 음절당 평균 2.454개로 너무 길어지는 문제가 있었고, 반대로 음절의 경우 음절당 평균 1개로 문자열을 짧게 표현할 수 있지만, 문자 집합의 크기가 1만이 넘어가는 문제가 있었기 때문이다. 하지만 종성만 별도로 분리하면 문자 집합의 크기가 427개(초성 19개 × 중성 21개 + 종성 27개)로 비교적 작으면서도 음절당 평균 길이도 1.454이 되어 메모리와 속도 양쪽의 균형을 잡을 수 있는 표현 방법이 된다.

이와 더불어 몇 가지 유용한 특징을 추가로 지닌다. 분석 과정에서 용언의 활용형에 대해서 원

형을 복원해야 하는 작업이 필요한데, 음절 단위에서는 모든 활용형이 다 다른 음절을 가지므로 활용형들 사이에서 공통 접두사[45]가 거의 존재하지 않는다. 반면 종성만 분리한 경우에서는 자소 단위에서와 마찬가지로 모든 활용형들에 대해 대체로 2~3개의 공통 접두사를 추출해내는 것이 가능하다.

또한 종성만으로 이뤄진 어미나 조사 등의 형태소가 결합하는 경우, 음절 단위 표현법에서는 음절에 종성이 붙는 것을 별도로 계산하는 절차(버리 + ㅁ → 버리ㅁ → 버림)가 필요하지만, 종성만 분리하는 표현 방법에서는 그대로 이어붙일 수 있다(버리 + ㅁ → 버리ㅁ).

표 2. 한글 표현 방법 비교

한글 표현 방법	음절(완성형)	자소(조합형)	종성만 분리	UTF8 Byte
설명	초성, 중성, 종성이 결합된 음절 하나를 최소 단위로 설정	초성, 중성, 종성을 전부 분리하여 각각을 최소 단위로 설정	초성, 중성은 결합하고 종성만 분리하여 각각을 최소 단위로 설정	유니코드(UTF8) 상에서의 한글 인코딩을 그대로 사용하고, Byte 단위로 분리
예시	글, 자	ㄱ,ㅡ,ㄹ,ㅈ,ㅏ	그,ㄹ,자	EA, B8, 80, EC, 9E, 90
문자 집합의 크기	11172	67	426	256
한글 1 음절의 평균 길이 [46]	1	2.454	1.454	3
활용형 간의 공통 접두사 (버리다 / 버림 / 버린 / 버렸다 / 버려서)	버	ㅂㅓㄹㅣ, ㅂㅓㄹㅕ	버리, 버려	EB B2 84 EB

이런 점을 고려하여 Kiwi에서는 내부 한글 표현 방법으로 종성만 분리하는 것을 채택했고, 표준으로 사용되는 한글 인코딩 방법인 유니코드와의 호환을 위해 형태소 분석 전에는 입력된 텍스트에서 종성만 분리하는 전처리 단계를, 분석 후에는 분리된 종성을 다시 결합하는 후처리 단계를 추가하였다.

3.1.2 품사 태그

표 3. Kiwi 에서 사용되는 품사 태그 목록

대분류	태그	설명
체언(N)	NNG	일반 명사
	NNP	고유 명사
	NNB	의존 명사
	NR	수사

	NP	대명사
용언(V)	VV	동사
	VA	형용사
	VX	보조 용언
	VCP	긍정 지시사(이다)
	VCN	부정 지시사(아니다)
	-R	규칙 활용*
	-I	불규칙 활용*
관형사	MM	관형사
부사(MA)	MAG	일반 부사
	MAJ	접속 부사
감탄사	IC	감탄사
조사(J)	JKS	주격 조사
	JKC	보격 조사
	JKG	관형격 조사
	JKO	목적격 조사
	JKB	부사격 조사
	JKV	호격 조사
	JKQ	인용격 조사
	JX	보조사
	JC	접속 조사
어미(E)	EP	선어말 어미
	EF	종결 어미
	EC	연결 어미
	ETN	명사형 전성 어미
	ETM	관형형 전성 어미
접두사	XPN	체언 접두사
접미사(XS)	XSN	명사 파생 접미사
	XSV	동사 파생 접미사
	XSA	형용사 파생 접미사
	XSM	부사 파생 접미사*
어근	XR	어근
부호, 외국어, 특수문자(S)	SF	종결 부호(. ! ?)
	SP	구분 부호(, / : ;)
	SS	인용 부호 및 괄호(' " ( ) [ ] < > { } — ‘ ’ “ ” ≪ ≫ 등)
	SSO	SS 중 여는 부호*
	SSC	SS 중 닫는 부호*
	SE	줄임표(…)
	SO	붙임표(- ~)
	SW	기타 특수 문자
	SL	알파벳(A-Z a-z)

	SH	한자
	SN	숫자(0-9)
	SB	순서 있는 글머리(가. 나. 1. 2. 가) 나) 등)*
분석 불능	UN	분석 불능*
웹(W)	W_URL	URL 주소*
	W_EMAIL	이메일 주소*
	W_HASHTAG	해시태그(#abcd)*
	W_MENTION	멘션(@abcd)*
	W_SERIAL	일련번호(전화번호, 통장번호, IP 주소 등)*
기타	Z_CODA	덧붙은 받침*
	USER0~4	사용자 정의 태그*

Kiwi에서는 형태소 품사 태그로 21세기 세종계획[47]의 품사 태그 집합을 채택하고 몇 가지 태그를 추가로 정의하여 사용하였다. 전체 태그는 표 2와 같고 이 중 추가로 정의한 태그에는 별표(*)를 붙였다. 세종계획의 품사 태그에서 사실상 거의 쓰이지 않는 태그인 분석 불능 범주(NA, NF, NV)를 제외하였고 웹 텍스트 분석 시의 편의성을 위해 URL 주소(W_URL), 이메일 주소(W_EMAIL), 해시태그(W_HASHTAG), 멘션(W_MENTION), 일련번호(W_SERIAL) 패턴에 대한 품사 태그를 추가하였다. 또한 웹 상에서는 '했어용', '했어욥' 처럼 어미 뒤에 받침을 덧붙여서 다양하게 변형된 말투를 자주 사용하는데, 여기서 덧붙은 받침을 분리할 때 사용하도록 Z_CODA라는 품사 태그도 정의하였다. 또 인용 부호 및 괄호의 시작과 끝을 구분할 수 있도록 SS 태그를 SSO, SSC로 세분화하였고, 순서 있는 글머리를 위한 품사 태그인 SB를 추가했다. 그리고 부사 파생 접미사를 위한 품사 태그 XSM을 추가로 정의하였다.

또한 규칙 활용 용언과 불규칙 활용 용언을 구분하기 위해 -R, -I 태그를 신설하였다. 이 두 태그는 단독으로 사용되는 게 아니라 용언 태그의 접미사로 쓰인다. 예를 들어 VV-R은 규칙활용 동사, VA-I는 불규칙활용 형용사를 가리킨다.

3.2 Kiwi의 형태소 분석 과정

앞서 설명한 것처럼 Kiwi는 크게 형태소 분리와 품사 부착 두 단계로 분석을 수행한다. 각 단계를 구체적으로 설명하면 다음과 같다. 먼저 형태소 분리 단계에서는 입력 텍스트를 형태소 후보 단위로 분리한다. 이 때 어휘집과 형태소 결합 규칙을 사용한다. 시스템은 초기화 단계에서 어휘집 내의 형태소들 중 결합 규칙에 의해 결합 가능한 것들을 미리 결합하여 어휘집에 추가로 등재한다. 그리고 어휘집을 트라이 자료 구조를 사용하여 저장해둔다. 따라서 실제 텍스트가 입력되는 시점에서는 어휘집만 사용해 탐색하여도 결합 규칙이 적용된 모든 후보를 탐색하게 되는 셈이다. 입력 텍스트는 트라이 자료 구조를 이용해 효율적으로 형태소 후보로 분리되고 분리된 후보들은 격자 구조로 변환된다.

격자 구조가 구축되면 이제 언어 모델을 통해 최적의 경로를 탐색할 차례이다. 시작 지점에서 끝 지점까지 이르는 모든 경로 중에서 확률 값이 가장 높은 경로를 선택하여 그 격자에 해당하는 품사 태그들을 각 형태소에 부착하면 품사 부착 단계가 완료된다. 최적 경로를 탐색할 때는 비터비(Viterbi) 알고리즘 혹은 빔 탐색(Beam Search) 알고리즘을 사용한다. 또한 각 경로를 탐색하

는 과정에서 규칙 기반으로 부적절한 띄어쓰기나 형태소 연속이 있는 경우 벌점을 추가하여 보정을 실시한다.

3.2.1 형태소 결합 규칙

한국어 맞춤법에서는 가능하면 형태소를 밝혀서 한국어를 적도록 규정하고 있으나 용언이 활용되면서 어미와 결합하면 그 형태소가 온전하게 드러나지 않는 일이 잦다. 따라서 형태소 분석기에서는 활용된 용언의 원형을 찾아서 어간과 어미를 분리하는 작업이 필수적이다. 다행히도 한국어 용언의 활용은 유형별로 규칙에 의해 정의될 수 있으므로 규칙과 이에 대한 일부 예외만 처리하면 쉽게 원형 복원이 가능하다.

Kiwi에서는 초기화 단계에서 규칙에 기반하여 결합 가능한 형태소들의 조합을 모두 탐색하여 이를 어휘집에 저장한다. 이 방법은 형태소 분석에 앞서 초기화에 소요되는 시간이 증가한다는 단점이 있으나, 사용자가 새 형태소를 사전에 추가한 경우 이 형태소의 활용형까지도 자동으로 형태 사전에 등재되므로 사용자의 요구에 맞춰 사전을 조작하는 게 더 용이하다는 장점이 있다. Kiwi에서 사용하는 형태소 결합 규칙 중 일부는 표 3과 같다.

표 4. Kiwi 의 형태소 결합 규칙 [48]

번호	규칙	설명
1	# 불규칙 활용 용언 PI E 르 어 르)(러	#으로 시작하는 줄은 주석이다. 먼저 두 알파벳은 품사 태그를 접두사를 가리킨다. 대부분 세종 품사 태그와 의미가 동일하다. 단 예외로 PI 는 불규칙 용언 전체를 가리키는데에 사용된다. 즉 PI E 는 왼쪽에는 불규칙 용언이 오고 오른쪽에는 어미가 오는 경우를 지정한다. 그 다음 줄부터는 형태소의 형태를 지정한다. 왼쪽에는 ‘르’로 끝나고, 오른쪽은 ‘어’로 시작하는 형태소가 올 때 ‘르러’라는 형태로 결합됨을 의미한다. 이 때 ‘)’ 기호는 왼쪽 형태소의 경계가 이 지점에서 끝남을, ‘(’기호는 오른쪽 형태소의 경계가 이 지점에서 시작함을 의미한다.
2	# 예외인 동사들 목록 VV E ^푸 어 퍼 따르 어 따(라) (후략)	이 규칙은 왼쪽에는 동사가 오고 오른쪽에는 어미가 오는 경우의 결합 방식을 지정한다. ^는 시작지점, $는 끝지점을 뜻하는 기호로, ‘^A’는 ‘A’로 시작하는 형태를, ‘A$’는 A 로 끝나는 형태를 뜻한다. 왼쪽 형태소에는 $가 항상 붙어 있으며, 오른쪽 형태소에는 ^가 항상 붙어 있는 것으로 간주하므로 규칙에서는 생략한다. 따라서 ‘^푸’는 형태가 ‘푸’와 온전히 일치하는 동사만을 지정한다. 그 다음 줄에서는 ‘…따르다’ 동사에 어미가 결합하는 경우를 지정한다. 이 규칙으로 따르다, 뒤따르다, 잇따르다 등의 결합 방식을 모두

		지정할 수 있다. 참고로 '…르다' 불규칙 용언의 결합 방식은 1 번 규칙에서 지정했지만 '따르다' 동사에는 규칙 활용 태그가 할당되어 있어 1 번 규칙에는 걸리지 않고 2 번 규칙을 적용하게 된다.
3	# '하다'와 어미의 결합 V,XSV,XSA E ^하 어 해,하)(여 ^하 다$ 타-1.1,)다-1 (후략)	이 규칙은 동사 및 동사/형용사 파생 접미사로 쓰이는 '하다'의 불규칙 활용을 다룬다. '…하다'에 '어…'가 결합될 경우 '하여', '해' 두 가지 표면형이 모두 가능하므로 이를 전부 기재해 주었다. 두번째 규칙에서는 '…하다'가 '…타' 혹은 '…다'로 축약되는 경우를 다룬다. 단 이 경우는 흔히 사용되는 표현은 아니므로 각 표면형에 -1.1, -1 점의 벌점을 부여하여 시스템이 '타' 혹은 '다'를 '하다'로 과도하게 복원하지 않도록 하였다.
4	# 보조 용언의 결합 VX E ㄹ [ㄴㅂㅅㄹ나-니바-비사-시오])\2 ㄹ ㅁ$ ㄺ [가나] 어 \1 (후략)	이 규칙은 보조용언에 어미가 결합하는 경우를 다룬다. 참고로 표면형에서 \1 은 왼쪽의 형태 전체를 가리키고, \2 은 오른쪽의 형태 전체를 가리킨다. 첫 번째 규칙은 왼쪽 형태소가 'ㄹ' 받침으로 끝나고 오른쪽 형태소가 'ㄴ', 'ㅂ', 'ㅅ', 'ㄹ' 받침이거나 초성이 'ㄴ'(나-니), 'ㅂ'(바-비), 'ㅅ'(사-시) 혹은 '오'로 시작하는 경우 표면형이 \2, 즉 오른쪽의 형태만 남길 것을 지정한다. 즉 왼쪽 형태소의 'ㄹ' 받침이 탈락하는 규칙을 정의한 것이다. 두번째 규칙은 'ㄹ' 받침과 'ㅁ' 어미가 결합하여 'ㄺ' 받침이 되는 것을 정의한다. 세번째 규칙은 왼쪽 형태소가 '가' 혹은 '나'로 끝나고 오른쪽에 '어'로 시작하는 어미가 올 경우 '어'는 없어지고 왼쪽의 형태만 남기도록 한다. 즉 모음이 축약되는 경우를 정의한다.

Kiwi에서 사용하는 결합 규칙은 총 20가지로, 각 규칙들은 적용 대상이 되는 품사와 그 형태를 되도록 한정적으로 정의하여 분석기 초기화 단계에서 결합 가능한 형태소 조합을 효율적으로 탐색할 수 있게 하였다.

3.2.2 어휘집 기반의 형태소 분리

형태소 분석기 초기화 단계에서 가능한 결합 규칙을 미리 적용해 어휘집에 등록해 두었으므로 실제 분석 시에는 트라이를 통해 텍스트 내에서 형태소 후보를 추출할 수 있다. 특히 아호 코라식(Aho-corasick) 알고리즘[49]을 사용하면 추출 과정을 크게 가속할 수 있다. 다만 모든 형태소가 어휘집에 등록되어 있을 수는 없기 때문에 미등록어를 다루기 위해 몇 가지 경험론적 보정 규칙을 도입하였다.

첫째는 패턴 기반 일치로 이는 특수 문자, 숫자, 로마자, 한자 등으로 구성된 문자열을 별도로

처리하기 위해 도입하였다. 예를 들어 세종 품사 태그에서는 임의의 숫자열에는 SN 태그를, 영단어에는 SL을 일괄적으로 부여하고 있다. 따라서 이런 문자열에 대해서는 별도로 어휘집 탐색 절차를 거치지 않고 패턴 일치 후 바로 태그를 부여하도록 하였다. 다만 419나 Rh+와 같이 특수문자로 구성되어 있더라도 고유명사로 쓰이는 단어에 대해서는 어휘집에서 처리하도록 하였다.

둘째는 규칙 기반의 오타 교정이다. 한국어에서 발생하는 오타나 맞춤법 오류의 유형은 다양하지만 그 중 많은 비율이 음가가 비슷한 글자나 글자열을 헷갈려 잘못 사용하는 것에 기인한다. 규칙 기반의 오타 교정은 한국어에서 통용되는 발음 규칙에 기반하여 어휘집 탐색 범위를 조절하여 오타에 대해서도 강건한 분석기를 구현할 수 있게 한다. '마갔다(막았다)'와 같은 입력을 예로 들자면, 받침의 자음과 초성의 ㅇ이 연쇄하는 경우 받침을 뒤로 넘겨 적는 변형이 가능하다고 규칙을 정의하면 실제 탐색 과정에서 시스템은 '마가'라는 문자열을 발견한 뒤 어휘집에서 '마가'뿐만 아니라 '막아'도 함께 탐색하도록 할 수 있다. 다만 이 경우 실제 '마가'를 의도하고 적은 문자열까지 '막아'로 잘못 교정될 수 있으므로 '마가'가 '막아'로 변형될 때는 어느 정도 벌점을 부여하여 과도 교정을 막았다. 오타 교정 규칙은 Kiwi의 소스 코드[50]에서 확인할 수 있다.

어휘집과 규칙을 통해 입력 문자열이 형태소 후보로 분할되었다면 언어 모델이 이를 효율적으로 탐색할 수 있게 최종적으로 격자(Lattice) 구조로 변환한다. 격자 구조에는 시작점과 끝점이 존재하며 그 사이에는 다양한 형태소 후보가 놓이게 된다. 따라서 시작점에서 끝점까지를 잇는 경로가 다양하게 존재하게 되는데 언어 모델은 이 중 가장 확률이 높은 경로를 찾는 역할을 수행한다.

3.2.3 언어 모델 기반의 품사 부착

Kiwi에서 사용하는 언어 모델은 앞서 언급한 것처럼 통계를 기반으로 한 n-gram 언어 모델이다. n-gram 언어 모델은 이전 $n-1$개의 단어 맥락을 이용하여 다음 단어가 등장할 확률을 계산하는 모델로 복잡한 연산 없이도 말뭉치 내에 존재하는 단어들의 빈도를 조사하는 것만으로 확률을 계산할 수 있어서 신경망 기반 언어 모델에 비해 효율적이다. 단 말뭉치 내에 등장한 적이 없는 단어열에 대해서는 확률을 계산할 수 없다는 희소 문제(sparsity problem)가 있고 이를 개선하기 위해 다양한 평탄화(smoothing) 기법이 제안되어 왔다. Kiwi에서는 현재 알려진 평탄화 기법 중 가장 품질이 좋은 것으로 알려진 Modified Kneser-Ney smoothing을 사용한 n-gram 언어 모델(KNLM)을 형태소열에 맞게 살짝 변형하여 사용한다. 구체적으로 Kiwi에서는 형태소열 $m_{1\cdots n-1}$이 주어졌을 때 다음 형태소로 m_n이 나타날 확률을 계산하기 위하여 말뭉치에서 빈도를 계산하는 방식 $N(m_0, m_1, \cdots, m_n)$을 다음과 같이 바꾸어 정의하였다.

$$N(m\_0) = c(m\_0)$$
$$N(m\_0, m\_1, \cdots, m\_n) = c(POS(m\_0), m\_1, \cdots, m\_n)$$

여기서 $c(m_0)$은 말뭉치에서 형태소 m_0이 등장한 빈도를 나타내고, $c(POS(m_0), m_1, \cdots, m_n)$은 형태소 m_0의 품사 태그와 동일한 형태소 아무것이나 등장한 이후 형태소열 $m_1, \cdots, m_n$이 차례로 등장한 빈도를 나타낸다. 즉, 길이가 2이상인 형태소열의 빈도를 탐색할 때 첫 번째 형태소는 품사 태그만 일치해도 전체가 일치한 것으로 간주한다는 점이 일반적인 KNLM과 다른 점이다. $n=3$인 경우를 예를 들자면 주어진 형태소 m_3 앞의 형태소 2개 m_1, m_2와 더불어 그 앞의 형태소의 품사 태그 $POS(m_0)$까지 고려하는 언어 모델이 된다. 따라서 배열의 길이가 4이므로 4-gram 언

어 모델의 일종이지만 실제 온전한 형태소는 3개이고 제일 앞의 1개는 품사태그만 고려하므로, 이를 편의상 3.5-gram 품사-형태소 언어 모델이라고 부를 수 있을 것이다. 모델을 이렇게 변형한 까닭은 n-gram 언어 모델은 n이 클수록 더 긴 문맥을 고려할 수 있어 성능이 향상되지만 동시에 과적합 문제와 희소 문제가 심화되고 모델을 저장하는 데에 필요한 메모리가 기하급수적으로 증가하기 때문이다. 그러나 품사-형태소 언어 모델은 일반적인 형태소 기반 n-gram 모델보다 1만큼 더 긴 문맥을 유지하면서도 모델 크기를 줄일 수 있고 또한 과적합 문제 및 희소 문제를 어느 정도 피할 수 있다.

3.2.4 장거리 의존성 해소를 위한 Skip-Bigram 언어 모델

Kiwi에서 기본적으로 채택한 n-gram 품사-형태소 언어 모델은 가까운 거리 내에 있는 형태소 간의 관계는 잘 계산해낼 수 있지만 멀리 떨어져 있는 형태소 간의 관계는 전혀 고려하지 못한다. 한국어에서 자주 쓰이는 문법 형태소인 조사와 어미는 바로 앞에 위치한 체언 혹은 용언과의 관계를 통해 쉽게 그 분포를 추정할 수 있기 때문에 문법적 문제는 n-gram 언어 모델로 비교적 쉽게 해결될 수 있다. 그러나 의미적 문제는 멀리 떨어진 형태소들 간의 관계를 고려해야만 풀 수 있는 경우가 많기 때문에 단순히 n-gram 언어 모델로는 풀기 어렵다. 특히 한국어는 어순이 비교적 자유로운 편이고 주어와 술어 사이에 목적어가 끼어드는 어순을 가지기에 장거리 의존성(Long-term dependency)이 발생하는 경우가 더욱 많다.

Kiwi는 이런 경우를 해결하기 위해 조금 더 개선된 언어 모델인 Skip-Bigram(SBG)을 고안하여 추가적으로 도입하였다. SBG은 Word2Vec[51]의 학습 방법 중 하나인 Skipgram의 아이디어를 발전시킨 것으로 기본적으로 KNLM의 확률 계산에 의존하지만 실질 형태소에 대해서는 앞의 문맥에 따라 확률을 보정하는 방식으로 작동한다. Word2Vec Skipgram 모델은 특정 단어로부터 일정한 거리 내의 있는 단어들이 그 어순과는 관계없이 그 단어의 등장에 영향을 준다는 가정을 바탕으로 한다. SBG에서도 이와 동일한 가정을 사용하며 구체적으로 SBG의 형태소 등장확률 P_{SBG}는 다음과 같이 계산된다.

$$P_{SBG}(m_n|m_0, m_1, \cdots, m_{n-1}) = P_{KN}(m_n|m_0, m_1, \cdots, m_{n-1}) \cdot \big(1 - d(m_n)\big) + \left(\sum_{i<n, m_i \in S} P_B(m_n|m_i)\right) \frac{1}{\sum_{i<n, m_i \in S} 1} \cdot d(m_n)$$

여기서 P_{KN}은 KNLM에 의한 형태소 등장 확률이며, $P_B(m_n|m_i)$는 실질 형태소 m_i가 왼쪽에 등장했을 때 형태소 m_n이 등장할 확률, S는 모든 실질 형태소의 집합, $d(m_n)$는 형태소 m_n의 등장 확률을 계산할 때 얼마나 P_B 확률을 반영할 것인지 조절하는 가중치를 뜻한다. 여기서 P_{KN}은 인접한 n개의 형태소만을 형태소의 순서를 고려해 확률 계산에 사용하지만, P_B는 왼쪽의 위치한 형태소라면 거리에 관계없이 모두 확률 계산에 사용한다. 다만 순서는 고려하지 않는다. P_B는 $R^{|S|\times|S|}$의 행렬로, d는 $R^{|S|}$의 벡터로 표현 가능하므로 SBG모델을 저장하는 데에는 전체 실질 형태소 개수의 제곱만큼의 추가 메모리가 필요하다. 다행히도 P_B를 나타내는 행렬은 대부분의 값이 0에 가까운 희소 행렬(sparse matrix)이므로 이를 효율적으로 저장하는 자료구조를 이용하면 모델이 사용하는 메모리를 효과적으로 줄일 수 있다. 먼저 학습이 완료된 KNLM과 SBG 학습에 사

용할 말뭉치가 주어진 경우, 신경망 학습에 흔히 사용하는 확률적 경사 하강법(Stochastic Gradient Descent)을 사용하여 SBG모델을 쉽게 학습할 수 있다.

3.2.5 규칙 기반 후처리 보정

품사-형태소 언어 모델 기반의 확률 점수는 모델이 적절한 형태소를 선택하는 데에 큰 역할을 하지만 때때로 통계 모델의 결과가 한국어 문법 규칙과 어긋나는 경우도 존재한다. 이를 개선하기 위해 Kiwi에서는 언어 모델 탐색 이후 마지막 단계로 규칙 기반의 보정을 시도한다. 사용하는 규칙은 띄어쓰기 벌점과 형태 벌점, 두 종류이다.

띄어쓰기 벌점은 왼쪽에 띄어쓰기가 요구되는 품사(조사, 어미가 아닌 품사)임에도 띄어쓰기가 없거나 반대로 왼쪽에 띄어쓰기가 없어야 하는 품사(조사와 어미)임에도 띄어쓰기가 있는 경우에 언어 모델 점수에 벌점을 부여하여 해당 탐색 결과가 선택될 확률을 낮춘다. 형태 벌점은 불규칙 활용 용언에 규칙 어미가 결합하거나 양성 모음 용언에 음성 모음 어미가 뒤따르거나 혹은 반대의 경우 등에 벌점을 부여한다.

이와 같이 문법 규칙상 허용되지 않는 형태소에 대해 언어 모델 탐색 과정에서 해당 경로를 아예 기각하지 않고 탐색 후에 벌점만 부여하는 것은 비문이나 오타가 섞인 문장에 대해서도 어느 정도 정확도를 보장하기 위해서이다. 만약 띄어쓰기 및 형태 규칙을 언어 모델 탐색 과정에 전에 적용한다면 '막었다'와 같은 표현을 '막/VV 었/EP 다/EP'로 분석할 수 없게 된다. 그러나 이를 후처리 과정에서 벌점만 부여하는 방식으로 처리한다면 해당 경로의 확률이 벌점을 받아 낮아졌을지라도 그보다 더 적절한 분석 경로가 없을 때에는 해당 경로가 선택될 수가 있어 '막었다'를 바르게 분석할 수 있다.

4. Kiwi 모델 학습과 평가

앞서 언급했듯 Kiwi는 형태소 분리 단계와 품사 부착 단계를 통해 형태소 분석을 실시한다. 형태소 분리 단계는 어휘집과 규칙 기반으로 작동하기에 말뭉치를 통한 모델 학습이 필요 없지만 언어 모델을 기반으로 하는 품사 부착 단계는 말뭉치를 통한 학습이 필요하다. 특히 KNLM과 SBG 학습에는 형태소 주석이 달린 형태 분석 말뭉치가 필요한데, 다행히도 21세기 세종 계획을 통해 약 1000만 어절의 형태 분석 말뭉치가 확보되었고, 2020년 국립국어원의 모두의 말뭉치 사업을 통해 추가로 300만 어절의 형태 분석 말뭉치가 구축되었다. Kiwi는 이 두 말뭉치를 활용해 어휘집을 추출하고 KNLM, SBG 모델을 학습하였다.

4.1 형태소 분석 정확도 평가

Kiwi의 품사 태그는 대체로 세종 품사 태그와 일치하지만 일부 수정된 태그들이 있기 때문에 기존의 말뭉치를 그대로 평가에 사용하기가 어렵다. 따라서 형태소 분석기의 성능을 측정하기 위해서 문어체 텍스트, 웹 텍스트, 오타가 포함된 웹 텍스트 세 종류의 평가 말뭉치를 구축하였

다.[52]

표 4는 KNLM 언어 모델의 구성에 따른 정확도와 속도를 비교한 결과이다. 일반적인 3-gram, 4-gram 언어 모델은 문어 텍스트의 정확도가 높은 반면 웹 텍스트의 정확도는 상대적으로 낮았다. 반대로 품사-형태소 언어 모델(KNLM 2.5-gram, KNLM 3.5-gram)은 문어 텍스트의 정확도가 살짝 낮지만 웹 텍스트나 오타에 더 강건한 모습을 보였다. 이는 세종 계획 말뭉치 및 모두의 말뭉치가 문어체에 치중되어 있기에 일반적인 n-gram 언어 모델로 이를 학습할 경우 문어체에 과적합된다는 것을 뒷받침한다. 추가로 3.2.2에서 설명한 오타 교정을 도입할 경우 문어체 정확도를 약간 희생하여 웹 텍스트에서 크게 정확도를 향상시킬 수 있음을 확인할 수 있다. Kiwi는 범용적인 형태소 분석기를 지향하므로 문어체에 치중되기 보다는 문어체와 웹 텍스트를 균형 있게 잘 분석할 수 있도록 3.5-gram의 품사-형태소 언어 모델을 기본 모델로 채택하였다.

표 5. n-gram 언어 모델의 정확도와 분석 속도 비교

이름	품사 태그 문맥 길이	형태소 문맥 길이	언어 모델의 크기(MB)	정확도(%)			분석 속도(KB/s)
				문어 텍스트	웹 텍스트	오타가 있는 웹 텍스트	
KNLM 3-gram	0	2	43.78	**94.61**	85.58	72.94	**155.02**
KNLM 4-gram	0	3	62.83	94.13	85.69	73.28	141.34
KNLM 2.5-gram	1	2	18.82	93.20	86.31	74.01	139.97
KNLM 3.5-gram	1	3	34.20	93.32	86.49	75.68	114.88
KNLM 3.5-gram (+ 오타교정)	1	3	34.20	92.63	**86.79**	**87.60**	38.12

4.2 모호성 해소 정확도 평가

각 형태소 분석기마다 채택한 품사 태그의 집합이 다르고 또 분석기마다 목표로 하는 분석 단위가 상이하기 때문에 4.1에서 구축한 평가 말뭉치로 서로 다른 분석기들의 정확도를 비교하는 것은 어렵다. 따라서 서로 다른 형태소 분석기를 공평하게 비교하기 위해 모호성 해소를 평가 척도로 사용했다. 모호성 해소 평가는 모호한 입력 문장에서 필수로 포함되어야 하는 형태소(정답 형태소) 하나를 선정하여 분석기가 이 형태소를 추출해냈는지를 확인하는 것으로 진행된다. 예를 들어 '길을 걸었다'는 문장이 바르게 형태소 분석되었다면 '걸다'가 아닌 '걷다'라는 동사가 분석 결과 내에 포함되어야 한다. 따라서 '걷다'가 분석 결과 내에 등장하면 해당 문장을 맞힌 것으로, 그렇지 않으면 틀린 것으로 판단한다. 이를 다양한 문장에 대해 수행하여 분석기가 문장을 맞힌 비율을 분석기의 모호성 해소 정확도 점수로 사용하였다. 이 평가를 위해서 총 4종류의 모호성 해소 말뭉치(규칙/불규칙 활용 동사 구분, 모호한 명사 구분, 동사/형용사 구분, 장거리 의존성)를 구축하였다.[53] 이 중 장거리 의존성 말뭉치는 먼 거리에 떨어진 형태소를 봐야만 정답 형태소를 바르게 뽑을 수 있는 문장들로 구성되어 있어 형태소 분석기의 원거리 맥락 인식 성능을 측정할

수 있다.

이 말뭉치를 통해 다양한 한국어 형태소 분석기별 모호성 해소 성능을 측정한 결과는 표 5와 같다. 먼저 KNLM을 사용한 Kiwi와 SBG를 사용한 Kiwi 간의 정확도 차이를 살펴보면 SBG 모델이 실제로 장거리 의존성 문제를 해결하는 데에 효과가 높다는 것을 확인할 수 있다. 또한 KNLM만을 사용한 경우에도 장거리 의존성을 제외한 나머지 항목에서 Kiwi가 타 형태소 분석기보다 정확도가 더 높다. 특히 딥러닝 모델을 사용한 바른이 나머지 분석기들보다 높지만 Kiwi보다는 살짝 낮은 정확도를 보인다는 점을 볼 때 잘 설정된 규칙과 고전적인 통계모델을 조합하는 접근법이 현 시점에도 유효하다고 할 수 있다.

표 6. 다양한 한국어 형태소 분석기들 간 모호성 해소 성능 비교

	규칙/불규칙 활용 동사 구분	모호한 명사 구분	동사/형용사 구분	장거리 의존성
Kiwi(KNLM)	82.1	**89.1**	**90.7**	58.1
Kiwi(SBG)	**89.6**	**89.1**	**90.7**	**77.4**
Komoran (v3.0)	46.3	54.5	40.7	41.9
MeCab (v2.1.1)	46.3	60.0	53.7	54.8
Kkma (v2.0)	52.2	70.9	46.3	41.9
Hannanum (v0.8.4)*	46.3	47.3	-	-
OKT (v2.1.0)*	46.3	52.7	-	-
Khaiii (v0.4)	55.2	67.3	61.1	48.4
Bareun (v2.2.1)	67.2	81.8	79.6	64.5

*Hannanum과 OKT는 분석 결과에서 동사와 형용사를 별도로 구분하지 않기에 일부 평가는 진행할 수 없었음

4.3 문장 분리 정확도 평가

형태소 분석기는 종결 어미를 찾아낼 수 있기 때문에 종종 문장을 분리하는 데에 사용되기도 한다. 글이 맞춤법에 맞게 잘 작성된 경우 마침표와 같은 문장 부호를 기준으로 쉽게 문장 분리가 가능하지만 그렇지 않은 경우 문장의 끝지점을 찾기 위해 다양한 문장 분리 규칙을 도입하거나 형태소 분석이 필수적이다. 특히 일부 종결 어미(-어, -지, -게 등)는 연결 어미와 형태가 동일하기 때문에 이 경우 형태소 분석기가 모호성을 바르게 해소해야만 문장을 정확하게 분리해낼 수 있다. 따라서 문장 분석 정확도 역시 형태소 분석기의 성능을 간접적으로 평가할 수 있는 지표이다.

문장 분리 평가 말뭉치[54]를 이용해 평가한 결과는 표 6과 같다. 베이스라인으로는 단순히 마침표를 기준으로 문장을 분리하는 시스템을 상정하였다. 일부 항목에 대해서는 Kiwi가 약간 낮은 문장 분리 정확도를 보이긴 했으나 명사 전성 어미를 구분하는 etn 말뭉치에서 높은 성능을 달성하여 평균 정확도는 Kiwi가 가장 앞서는 것으로 확인되었다. 분석 속도도 주목할 만하다. 단순히 규칙 기반으로 분리를 실시하는 Baseline과 Hannanum, OKT을 제외하면 Kiwi가 가장 빠른 속도를 달성하여 경량 통계 모델을 사용한 것이 효율성 면에서도 효과가 있음을 알 수 있다.

표 7. 문장 분리 평가 성능 비교

	Normalized F1						분석속도 (문장/s)
	blogs	etn	nested	tweets	wikipedia	평균	
Baseline	59.88	59.85	75.99	61.80	76.37	66.78	776190.4
Kiwi	83.81	**85.66**	91.44	78.98	97.10	**87.40**	481.5
Hannanum (v0.8.4)	62.51	59.85	82.03	66.92	97.28	73.72	2585.0
OKT (v2.1.0)	62.16	59.85	83.83	66.19	76.35	69.68	3124.7
KSS[55](+PeCab) (v5.2.0)	**88.87**	63.84	**92.06**	80.77	98.80	84.87	1.9
KSS(+MeCab) (v5.2.0)	**88.87**	63.84	**92.06**	**81.37**	**100.00**	85.23	436.3
Bareun (v2.2.1)	56.18	51.21	84.13	51.33	91.94	83.69	215.0

5. Kiwi 활용 사례

Kiwi의 초기 버전은 정확도나 사용성 측면에서 부족한 점이 많아서 실제로 연구나 NLP 시스템에 사용된 사례는 없는 것으로 보인다. 그러나 여러 차례의 업데이트로 정확도를 개선한 덕분에 2020년대에 들어 Kiwi를 시스템 개발에 사용하는 사례가 등장하기 시작했고 현재는 200건 이상의 GitHub 저장소뿐만 아니라 NLP, 텍스트 마이닝, 언어학, 인문학 등의 연구 분야에서 다양하게 사용되고 있다. 대표적인 사례를 정리해보면 다음과 같다.

5.1 텍스트 마이닝 및 자연어처리 분야

텍스트 마이닝 분야에서 Kiwi는 주로 분석을 위한 핵심 키워드를 추출하는 단계에 쓰이고 있다. 이렇게 추출된 키워드를 그 자체로 내용 분석에 사용하기도 하고 데이터 정제[56]에 쓰기도 한다. 빈도 분석과 단어 임베딩[57], 동시출현 분석과 네트워크 분석[58], 토픽 모델링[59] 등 다양한 마이닝 기법의 입력 데이터로 사용한 연구도 있다.

NLP 분야에서는 주로 한국어 입력 텍스트를 모델이 다루기 좋은 형태로 전처리하기 위한 용도로 사용되고 있다. 몇 가지 사례를 들어보자면 도서 주제 자동분류모델 연구[60]나 우울증 진단 다중 레이블 분류모델[61], 유튜브 악플 자동분류모델 연구[62] 등이 있다. 언어 모델이 생성한 텍스트를 평가하거나 후처리 하기 위해 Kiwi를 사용한 사례도 있다. 김아리와 김진현[63]은 모델이 생성한 텍스트를 평가할 때 Kiwi로 형태소 분석을 수행하면 타 분석기를 사용한 것보다 사람이 평가한 것과 일치도가 더욱 높은 결과를 얻을 수 있음을 보였다. 김태완[64]은 GPT-4가 생성한 텍스트에서 특정 키워드만 추출하기 위한 후처리 단계에 Kiwi를 사용했다.

오픈소스 프로젝트 중에 흥미로운 것으로는 Politely[65]가 있다. 이는 머신러닝과 규칙 기반으로 한국어 텍스트의 어체를 공손하게 바꿔주는 Python 패키지로 초기에는 타 분석기를 기반으로 개발되었으나 후에 Kiwi로 교체한 후 성능 개선이 확인되어 현재는 Kiwi의 형태소 분석 기능과 형태소 합성 기능을 활용하고 있다.

5.2 인문학 분야

인문학 분야에서도 Kiwi를 사용한 연구가 등장하고 있다. 특히 언어 사용 양상 그 자체가 연구의 대상이 되는 언어학에서 다양하게 사용되고 있다. 한국어 학습자의 정형 표현 사용 양상을 비교한 연구[66]나 한국어와 우즈베크어의 부사격 조사의 대응 양상을 연구한 사례[67], 한국어와 일본어 간의 외래어 사용 양상을 비교한 연구[68], GPT-4 모델과 인간의 한국어 사용 양상을 연구한 것[69] 등이 대표적이다.

신문이나 문학 작품을 분석하기 위해 형태소 분석기를 사용한 연구도 존재한다. 근대 소설 텍스트를 분석하여 '우리'라는 단어의 의미에 대해 분석을 시도한 연구[70], 1920년대 여성시인의 작품을 분석한 연구[71], 조선일보 기사를 분석하여 민족 담론의 의미 변화를 추적한 연구[72] 등이 대표적이다. 인문학 분야에 형태소 분석기를 사용한 사례는 아직 많지는 않지만 국내 디지털 인문학 연구가 활성화되면서 앞으로 Kiwi를 비롯한 오픈소스 형태소 분석기를 활용한 연구는 더욱 늘어날 것으로 예상된다.

6. 결론

어휘집과 규칙 기반의 형태소 분리와 Modified Knerser-ney smoothing가 적용된 품사-형태소 언어 모델과 Skip-Bigram 기반의 품사 부착을 통해 Kiwi는 빠르면서도 범용적으로 높은 정확도를 달성할 수 있었다. 덕분에 Kiwi는 NLP, 텍스트 마이닝, 언어학, 인문학 등 다양한 분야에서 활용되고 있지만 아직 개선해야 할 부분이 많이 남아있다.

제일 대표적인 것은 미등록어 및 신조어에 대한 성능 고도화이다. 분야에 맞춰 사용자 사전에 미등록어 및 신조어를 등록하는 식으로 현재 버전에서도 미등록어 문제에 대응할 수 있지만 이는 품이 크게 드는 작업이다. 최근 딥러닝 기반 접근법에서 시도되는 것처럼 처음 보는 단어일지라도 주변 문맥을 통해 적절하게 이를 처리할 수 있다면 새로운 분야의 텍스트를 형태소 분석하는 데에 크게 도움이 될 것이다.

또한 한국어 방언에 대한 지원이 부족한 점 역시 앞으로 개선해 나가야 할 부분이다. 형태소 분석기 구축 단계에서 의도적으로 방언을 배제하지는 않았지만 학습에 사용된 말뭉치가 표준어 중심적이었기 때문에 학습 결과로 산출된 언어 모델에서 자연스레 방언이 배제되어 방언에 대한 분석 정확도가 크게 떨어지는 문제가 있다. 이 부분을 보완해 나간다면 다양한 한국어 문학 작품을 분석할 때 크게 도움이 될 것이다.

이 밖에도 형태소 분석기에 포함되면 유용할 기능들이 분명 더 많이 있을 것이다. 오픈소스 형태소 분석기인만큼 오픈소스의 특성을 살려 다양한 연구자들이 이를 자유롭게 이용하고 의견을 모으며 필요한 기능들을 힘을 합쳐 꾸준히 개발해 나간다면 Kiwi가 앞으로 더 유용한 한국어 분석 도구가 될 것이라 확신한다.

---

[1] 김흥규(2004). <21 세기 세종계획 국어 기초자료 구축>. 문화관광부. 126.

[2] 주로 SentencePiece, WordPiece, Byte Pair Encoding 와 같은 토큰화 기법이 사용되며, 이런 토큰화 기법에서는 자주 등장하는 문자열을 하나의 토큰으로 묶는 방식을 사용한다. 이렇게 묶인 토큰이 우연히 형태소 단위와 일치할 수도 있지만 대체로 형태소 경계와 일치하지 않는 분할이 이뤄진다.

[3] Kim, Boseop; Kim, HyoungSeok; Lee, Sang-Woo; Lee, Gichang; Kwak, Donghyun; Jeon, Dong Hyeon; Park, Sunghyun; Kim, Sungju; Kim, Seonhoon; Seo, Dongpil. (2021, November). “What Changes Can Large-scale Language Models Bring? Intensive Study on HyperCLOVA: Billions-scale Korean Generative Pretrained Transformers.” Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing (pp. 3405-3424). https://doi.org/10.18653/v1/2021.emnlp-main.274

[4] Park, Kyubyong; Lee, Joohong; Jang, Seongbo; Jung, Dawoon. (2020, December). “An Empirical Study of Tokenization Strategies for Various Korean NLP Tasks.” Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing (pp. 133-142). https://aclanthology.org/2020.aacl-main.17/

[5] Hofmann, Valentin; Schuetze, Hinrich; Pierrehumbert, Janet. (2022). “An Embarrassingly Simple Method to Mitigate Undesirable Properties of Pretrained Language Model Tokenizers.” Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (pp. 385–393). https://doi.org/10.18653/v1/2022.acl-short.43

[6] 김성용(1987). “Tabular parsing 방법과 접속 정보를 이용한 한국어 형태소 분석기”. 한국과학기술원 석사 학위논문. http://hdl.handle.net/10203/33721

[7] 권오욱, 정유진, 김미영, 류동원, 이문기, 이종혁(1999). “음절단위 CYK 알고리즘에 기반한 형태소 분석기 및 품사태거”. <한국정보과학회 언어공학연구회 학술발표 논문집>. 76-87. https://www.riss.kr/link?id=A101722372

[8] 최재혁, 이상조(1993). “양방향 최장일치법에 의한 한국어 형태소 분석기에서의 사전 검색 횟수 감소 방안”. <정보과학회논문지> 20(10). 1497-1507. https://www.riss.kr/link?id=A82292437

[9] 양승현, 김영섭(2000). “부분 어절의 기분석에 기반한 고속 한국어 형태소 분석 방법”. <정보과학회논문지: 소프트웨어 및 응용> 27(3). 290-301. https://www.riss.kr/link?id=A82294456

[10] Lim, Heui-Suk; Lee, Sang-Zoo; Rim, Hae-Chang. (1995). “An efficient Korean morphological analysis using exclusive information.” Proceedings of the 1995 International Conference on Computer Processing of Oriental Languages (pp. 225-258).

[11] Kim, Jae-Hoon; Jang, Byung-Gyu; Kim, Gil Chang; Seo, Jungyun. (1995). “Morphological ambiguity reduction using subsumption relation in Korean.” Proceedings of the Natural Language Processing Pacific Rim Symposium (NLPRS’95).

[12] Lee, Gary Geunbae; Cha, Jeongwon; Lee, Jong-Hyeok. (2002). “Syllable-pattern-based unknown-morpheme segmentation and estimation for hybrid part-of-speech tagging of Korean”. Computational Linguistics, 28(1), 53-70. https://doi.org/10.1162/089120102317341774

[13] Lee, Do-Gil; Hae-Chang Rim. (2009). “Probabilistic modeling of Korean morphology.” IEEE transactions on audio, speech, and language processing, 17(5), 945-955. https://doi.org/10.1109/TASL.2009.2019922

[14] 이재성(2011). “한국어 형태소 분석을 위한 3 단계 확률 모델”. <정보과학회논문지: 소프트웨어 및 응용> 38(5). 257-268. https://www.riss.kr/link?id=A82599933

[15] Kudo, Taku; Yamamoto, Kaoru; Matsumoto, Yuji. (2004, July). “Applying conditional random fields to Japanese morphological analysis”. Proceedings of the 2004 conference on empirical methods in natural language processing (pp. 230-237). https://aclanthology.org/W04-3230/

[16] Kudo, Taku. “MeCab”. https://sourceforge.net/projects/mecab/

[17] 이용운. “은전한닢 프로젝트를 소개합니다”. https://eunjeon.blogspot.com/2013/02/blog-post.html

[18] 나승훈, 김창현, 김영길(2014). “래티스상의 구조적 분류에 기반한 한국어 형태소 분석 및 품사 태깅”. <정보과학회논문지: 소프트웨어 및 응용> 41(7). 523-532. https://www.riss.kr/link?id=A100050808

[19] 심광섭(2011). “형태소 분석기 사용을 배제한 음절 단위의 한국어 품사 태깅”. <인지과학> 22(3). 327-345. https://www.doi.org/10.19066/COGSCI.2011.22.3.005

[20] 서대룡, 정유진, 강인호(2017). “오타에 강건한 자모 조합 임베딩 기반 한국어 품사 태깅”. <한국어정보학회 학술대회>. 203-208. https://koreascience.kr/article/CFKO201712470015328.page

[21] 임재수. “Khaiii”. https://github.com/kakao/khaiii

[22] 한경은, 백슬예, 임재수(2017). “공개와 협업을 통한 세종 형태 분석 말뭉치 오류 개선 방법”. <한국어정보학회 학술대회> 228-232. https://koreascience.kr/article/CFKO201712470015351.page

[23] 이건일, 이의현, 이종혁(2017). “Sequence-to-sequence 기반 한국어 형태소 분석 및 품사 태깅”. <정보과학회논문지> 44(1). 57-62. https://www.riss.kr/link?id=A102592980

[24] 최병서, 이익훈, 이상구(2020). “신조어 및 띄어쓰기 오류에 강인한 시퀀스-투-시퀀스 기반 한국어 형태소 분석기”. <정보과학회논문지> 47(1). 70-77. http://dx.doi.org/10.5626/JOK.2020.47.1.70

[25] Devlin, Jacob; Chang, Ming-Wei; Lee, Kenton; Toutanova, Kristina. (2018). “Bert: Pre-training of deep bidirectional transformers for language understanding”. arXiv preprint arXiv:1810.04805. https://doi.org/10.48550/arXiv.1810.04805

[26] 민진우, 나승훈, 신종훈, 김영길(2019). “BERT 에 기반한 Subword 단위 한국어 형태소 분석”. <제 31 회 한글 및 한국어 정보처리 학술대회>, 37-40. https://koreascience.kr/article/CFKO201930060817853.page

[27] 박천음, 이창기, 김현기(2019). “BERT 기반 LSTM-CRF 모델을 이용한 한국어 형태소 분석 및 품사 태깅”. <제 31 회 한글 및 한국어 정보처리 학술대회>. 34-36. https://koreascience.kr/article/CFKO201930060574803.page

[28] 이재성, 박재득, 차건회, 박세영(1999). “형태소분석기 및 품사 태거 평가대회 (MATEC99) 개요”. <한국정보과학회 언어공학연구회 학술발표 논문집>, 13-22. https://www.riss.kr/link?id=A101722255

[29] 이운재, 김선배, 김길연, 최기선(1999). “모듈화된 형태소 분석기의 구현”. <한국정보과학회 언어공학연구회 : 학술대회논문집>. 123-136. https://koreascience.kr/article/CFKO199929013520621.page

[30] KAIST Semantic Web Research Center. “HanNanum - SWRC”. https://web.archive.org/web/20111020055238/http://semanticweb.kaist.ac.kr/home/index.php/HanNanum

[31] 이동주, 연종흠, 황인범, 이상구 (2010). “꼬꼬마: 관계형 데이터베이스를 활용한 세종 말뭉치 활용 도구”. <정보과학회논문지: 컴퓨팅의 실제 논문지> 16(11). 1046-1050. https://www.riss.kr/link?id=A82497594

[32] 서울대학교 Intelligent Data Systems 연구실. “꼬꼬마 한국어 형태소 분석기”. http://kkma.snu.ac.kr/documents/index.jsp

[33] 수명. “Apache Lucene/Solr (+ML/AI) 커뮤니티”. http://cafe.naver.com/korlucene

[34] 김상준. “daon”. https://github.com/rasoio/daon

[35] 최석재. “RHINO”. https://github.com/SukjaeChoi/RHINO

[36] 신준수, 박정환, 이근호. “komoran”. https://github.com/shineware/KOMORAN

[37] 은전한닢. “mecab-ko-dic”. https://bitbucket.org/eunjeon/mecab-ko-dic

[38] 은전한닢. “seunjeon”. https://bitbucket.org/eunjeon/seunjeon

[39] 고현웅. “PeCab”. https://github.com/hyunwoongko/pecab

[40] 이민철. “Kiwi”. https://github.com/bab2min/Kiwi

[41] open-korean-text. “open-korean-text”. https://github.com/open-korean-text/open-korean-text

[42] 임재수. “Khaiii”. https://github.com/kakao/khaiii

[43] 바른팀, “바른”. https://bareun.ai/

[44] Chen, Stanley F; Goodman, Joshua. (1999). “An empirical study of smoothing techniques for language modeling.” Computer Speech & Language, 13(4), 359-394. https://doi.org/10.1006/csla.1999.0128

[45] 컴퓨터 공학에서 공통 접두사(Common Prefix)는 여러 문자열 사이에서 공통되는 접두사(시작 문자열)를 뜻한다. 예를 들어 ABCD 와 ABE 의 공통 접두사는 AB 이다.

[46] 이 통계는 국립국어원 모두의 말뭉치 중 문어 말뭉치를 이용하여 계산되었음

[47] 김흥규(2004). <21 세기 세종계획 국어 기초자료 구축>. 문화관광부. 126.

[48] 결합 규칙 전체는 Kiwi github repository 에서 확인할 수 있다. https://github.com/bab2min/Kiwi/blob/707e51b722aca2d6ca4fb8be94a5a8117c76383d/ModelGenerator/combiningRule.txt

[49] Aho, Alfred V; Corasick, Margaret J. (1975). “Efficient string matching: an aid to bibliographic search”. Communications of the ACM, 18(6), 333-340. https://doi.org/10.1145/360825.360855

[50] https://github.com/bab2min/Kiwi/blob/37bfa605dcabed0bbfd1843d4ded46a43140e621/src/TypoTransformer.cpp#L470

[51] Mikolov, Tomas; Sutskever, Ilya; Chen, Kai; Corrado, Greg S; Dean, Jeff. (2013). “Distributed representations of words and phrases and their compositionality”. Advances in neural information processing systems, 26. https://doi.org/10.48550/arXiv.1310.4546

[52] 평가 말뭉치는 https://github.com/bab2min/Kiwi/tree/main/eval_data 에서 확인할 수 있음

[53] 모호성 해소 평가 말뭉치와 스크립트는 https://github.com/bab2min/kiwipiepy/tree/main/benchmark/disambiguate 에서 확인할 수 있음

[54] 문장 분리 평가 말뭉치와 스크립트는 https://github.com/bab2min/kiwipiepy/tree/main/benchmark/sentence_split 에서 확인할 수 있음

[55] 형태소 분석 결과에 후처리 규칙을 적용해 문장을 분리하는 오픈소스 문장 분리 패키지이다. 고현웅, 박상길. “Kss: A Toolkit for Korean sentence segmentation”. https://github.com/hyunwoongko/kss

[56] 유용상, 정민화, 이승민, 송민(2023). “KOMUChat: 인공지능 학습을 위한 온라인 커뮤니티 대화 데이터셋 연구”. <지능정보연구> 29(2). 219-240. https://www.riss.kr/link?id=A108639227

[57] 이희진, 박소현, 이유나, 한수희, 김바로(2023). “텍스트 마이닝을 통한 한중 웹소설 플랫폼 비교 분석”. <인문사회과학연구> 31(1). 314-343. https://www.riss.kr/link?id=A108488760

[58] 구자석, 박민수, 이경민(2022). “사회 연결망 분석을 통한 기계학회 신뢰성 연구 동향”. <대한기계학회 춘추학술대회>. 599-604. https://www.riss.kr/link?id=A108424057

[59] 조호수, 장문경, 류민호(2021). “코로나 19 팬데믹 상황에서 살펴본 민간 주도 정보제공의 역할 분석”. <한국콘텐츠학회논문지> 21(4). 1-13. https://www.doi.org/10.5392/JKCA.2021.21.04.001

[60] 김무성(2020). “온라인 서점 도서 데이터를 활용한 도서 주제 자동분류모델 구현”. <한국정보과학회 학술발표논문집>. 1484-1486. https://www.riss.kr/link?id=A107279386

[61] 양현동, 오하영(2023). “단락 벡터 기반 DSM-5 우울 진단 멀티 라벨 모델”. <한국정보통신학회논문지> 27(10). 1201-1207. https://www.riss.kr/link?id=A108809190

[62] 이신행, 이주연, 조민정, 박태강(2022). “기계학습 기반 유튜브 악플 분석:“사이버렉카” 에 달린 댓글의 어휘적 특성”. <한국디지털콘텐츠학회논문지> 23(6). 1115-1122. http://dx.doi.org/10.9728/dcs.2022.23.6.1115

[63] Kim, Ahrii and Kim, Jinhyeon. (2022, May). “Vacillating human correlation of sacrebleu in unprotected languages”. Proceedings of the 2nd Workshop on Human Evaluation of NLP Systems (HumEval). 1-15. https://doi.org/10.18653/v1/2022.humeval-1.1

[64] Kim, Taewan; Bae, Seolyeong; Kim, Hyun Ah; Lee, Su-woo; Hong, Hwajung; Yang, Chanmo; Kim, Young-Ho. (2023). “MindfulDiary: Harnessing Large Language Model to Support Psychiatric Patients' Journaling”. arXiv preprint arXiv:2310.05231. https://doi.org/10.48550/arXiv.2310.05231

[65] 김유빈. “Politely”. https://github.com/eubinecto/politely

[66] 정미경(2023). “한국어 학습자의 정형 표현 사용 양상 연구-한국어 학습자 말뭉치와 모두의 말뭉치의 비교를 중심으로”. <돈암어문학> 44. 291-326. http://dx.doi.org/10.17056/donam.2023.44..291

[67] 주은진, 굴하요(2023). “한국어 부사격 조사'에'의 우즈베크어 대응 양상 연구: 한국어-우즈베크어 병렬 말뭉치를 기반으로”. <번역학연구> 24(3). 563-590. http://dx.doi.org/10.15749/jts.2023.24.3.018

[68] 黃秀智(2023). 新聞を利用した日韓外来語の通時的対照研究. <동서인문학> 64. 115-140. https://www.riss.kr/link?id=A108495512

[69] 박서윤, 강예지, 강조은, 김유진, 이재원, 정가연, 최규리, 김한샘(2024). “GPT-4 를 활용한 인간과 인공지능의 한국어 사용 양상 비교 연구”. <국어국문학> (206). 5-47. https://www.kci.go.kr/kciportal/ci/sereArticleSearch/ciSereArtiView.kci?sereArticleSearchBean.artiId=ART003069022

[70] 徐在玄, 金炳俊, 金民雨, 朴素晶(2021). “멀리서 읽는 “우리”—Word2Vec, N-gram 을 이용한 근대 소설 텍스트 분석”. <대동문화연구> 115, 349-386. http://dx.doi.org/10.18219/ddmh..115.202109.349

[71] 임수경(2022). “1980 년대 여성시인의 시어'속'비교 연구-최승자, 김혜순의 시집을 중심으로”. <동아인문학> 58. 151-182. http://doi.org/10.52639/JEAH.2022.03.58.151

[72] 김병준, 전봉관(2023). “민족, 국민, 국가-시계열 워드 임베딩을 활용한 조선일보 기사의 민족 담론 의미 변동 추적 (1920~ 40)”. <현대소설연구> (90). 5-38. https://www.riss.kr/link?id=A108640991

참고문헌

고현웅, 박상길. “Kss: A Toolkit for Korean sentence segmentation”. https://github.com/hyunwoongko/kss

고현웅. “PeCab”. https://github.com/hyunwoongko/pecab

구자석, 박민수, 이경민(2022). “사회 연결망 분석을 통한 기계학회 신뢰성 연구 동향”. <대한기계학회 춘추학술대회>. 599-604. https://www.riss.kr/link?id=A108424057

권오욱, 정유진, 김미영, 류동원, 이문기, 이종혁(1999). “음절단위 CYK 알고리즘에 기반한 형태소 분석기 및 품사태거”. <한국정보과학회 언어공학연구회 학술발표 논문집>. 76-87. https://www.riss.kr/link?id=A101722372

김무성(2020). “온라인 서점 도서 데이터를 활용한 도서 주제 자동분류모델 구현”. <한국정보과학회 학술발표논문집>. 1484-1486. https://www.riss.kr/link?id=A107279386

김병준, 전봉관(2023). “민족, 국민, 국가-시계열 워드 임베딩을 활용한 조선일보 기사의 민족 담론 의미 변동 추적 (1920~ 40)”. <현대소설연구> (90). 5-38. https://www.riss.kr/link?id=A108640991

김상준. “daon”. https://github.com/rasoio/daon

김성용(1987). “Tabular parsing 방법과 접속 정보를 이용한 한국어 형태소 분석기”. 한국과학기술원 석사 학위논문. http://hdl.handle.net/10203/33721

김유빈. “Politely”. https://github.com/eubinecto/politely

김흥규(2004). <21 세기 세종계획 국어 기초자료 구축>. 문화관광부. 126.

나승훈, 김창현, 김영길(2014). “래티스상의 구조적 분류에 기반한 한국어 형태소 분석 및 품사 태깅”. <정보과학회논문지: 소프트웨어 및 응용> 41(7). 523-532. https://www.riss.kr/link?id=A100050808

민진우, 나승훈, 신종훈, 김영길(2019). “BERT 에 기반한 Subword 단위 한국어 형태소 분석”. <제 31 회 한글 및 한국어 정보처리 학술대회>, 37-40. https://koreascience.kr/article/CFKO201930060817853.page

박서윤, 강예지, 강조은, 김유진, 이재원, 정가연, 최규리, 김한샘(2024). “GPT-4 를 활용한 인간과 인공지능의 한국어 사용 양상 비교 연구”. <국어국문학> (206). 5-47. https://www.kci.go.kr/kciportal/ci/sereArticleSearch/ciSereArtiView.kci?sereArticleSearchBean.artiId=ART003069022

박천음, 이창기, 김현기(2019). “BERT 기반 LSTM-CRF 모델을 이용한 한국어 형태소 분석 및 품사 태깅”. <제 31 회 한글 및 한국어 정보처리 학술대회>. 34-36. https://koreascience.kr/article/CFKO201930060574803.page

서대룡, 정유진, 강인호(2017). “오타에 강건한 자모 조합 임베딩 기반 한국어 품사 태깅”. <한국어정보학회 학술대회>. 203-208. https://koreascience.kr/article/CFKO201712470015328.page

서울대학교 Intelligent Data Systems 연구실. “꼬꼬마 한국어 형태소 분석기”. http://kkma.snu.ac.kr/documents/index.jsp

徐在玄, 金炳俊, 金民雨, 朴素晶(2021). “멀리서 읽는 “우리”—Word2Vec, N-gram 을 이용한 근대 소설 텍스트 분석”. <대동문화연구> 115, 349-386. http://dx.doi.org/10.18219/ddmh..115.202109.349

수명. “Apache Lucene/Solr (+ML/AI) 커뮤니티”. http://cafe.naver.com/korlucene

신준수, 박정환, 이근호. “komoran”. https://github.com/shineware/KOMORAN

심광섭(2011). “형태소 분석기 사용을 배제한 음절 단위의 한국어 품사 태깅”. <인지과학> 22(3). 327-345. https://www.doi.org/10.19066/COGSCI.2011.22.3.005

양승현, 김영섬(2000). “부분 어절의 기분석에 기반한 고속 한국어 형태소 분석 방법”. <정보과학회논문지: 소프트웨어 및 응용> 27(3). 290-301. https://www.riss.kr/link?id=A82294456

양현동, 오하영(2023). “단락 벡터 기반 DSM-5 우울 진단 멀티 라벨 모델”. <한국정보통신학회논문지> 27(10). 1201-1207. https://www.riss.kr/link?id=A108809190

유용상, 정민화, 이승민, 송민(2023). “KOMUChat: 인공지능 학습을 위한 온라인 커뮤니티 대화 데이터셋 연구”. <지능정보연구> 29(2). 219-240. https://www.riss.kr/link?id=A108639227

은전한닢. “mecab-ko-dic”. https://bitbucket.org/eunjeon/mecab-ko-dic

은전한닢. “seunjeon”. https://bitbucket.org/eunjeon/seunjeon

이건일, 이의현, 이종혁(2017). “Sequence-to-sequence 기반 한국어 형태소 분석 및 품사 태깅”. <정보과학회논문지> 44(1). 57-62. https://www.riss.kr/link?id=A102592980

이동주, 연종흠, 황인범, 이상구 (2010). “꼬꼬마: 관계형 데이터베이스를 활용한 세종 말뭉치 활용 도구”. <정보과학회논문지: 컴퓨팅의 실제 논문지> 16(11). 1046-1050. https://www.riss.kr/link?id=A82497594

이민철. “Kiwi”. https://github.com/bab2min/Kiwi

이신행, 이주연, 조민정, 박태강(2022). “기계학습 기반 유튜브 악플 분석:“사이버렉카” 에 달린 댓글의 어휘적 특성”. <한국디지털콘텐츠학회논문지> 23(6). 1115-1122. http://dx.doi.org/10.9728/dcs.2022.23.6.1115

이용운. “은전한닢 프로젝트를 소개합니다”. https://eunjeon.blogspot.com/2013/02/blog-post.html

이운재, 김선배, 김길연, 최기선(1999). “모듈화된 형태소 분석기의 구현”. <한국정보과학회 언어공학연구회 : 학술대회논문집>. 123-136. https://koreascience.kr/article/CFKO199929013520621.page

이재성(2011). “한국어 형태소 분석을 위한 3 단계 확률 모델”. <정보과학회논문지: 소프트웨어 및 응용> 38(5). 257-268. https://www.riss.kr/link?id=A82599933

이재성, 박재득, 차건회, 박세영(1999). “형태소분석기 및 품사 태거 평가대회 (MATEC99) 개요”. <한국정보과학회 언어공학연구회 학술발표 논문집>, 13-22. https://www.riss.kr/link?id=A101722255

이희진, 박소현, 이유나, 한수희, 김바로(2023). “텍스트 마이닝을 통한 한중 웹소설 플랫폼 비교 분석”. <인문사회과학연구> 31(1). 314-343. https://www.riss.kr/link?id=A108488760

임수경(2022). “1980 년대 여성시인의 시어'속'비교 연구-최승자, 김혜순의 시집을 중심으로”. <동아인문학> 58. 151-182. http://doi.org/10.52639/JEAH.2022.03.58.151

임재수. “Khaiii”. https://github.com/kakao/khaiii

정미경(2023). “한국어 학습자의 정형 표현 사용 양상 연구-한국어 학습자 말뭉치와 모두의 말뭉치의 비교를 중심으로”. <돈암어문학> 44. 291-326. http://dx.doi.org/10.17056/donam.2023.44..291

조호수, 장문경, 류민호(2021). “코로나 19 팬데믹 상황에서 살펴본 민간 주도 정보제공의 역할 분석”. <한국콘텐츠학회논문지> 21(4). 1-13. https://www.doi.org/10.5392/JKCA.2021.21.04.001

주은진, 굴하요(2023). “한국어 부사격 조사'에'의 우즈베크어 대응 양상 연구: 한국어-우즈베크어 병렬 말뭉치를 기반으로”. <번역학연구> 24(3). 563-590. http://dx.doi.org/10.15749/jts.2023.24.3.018

최병서, 이익훈, 이상구(2020). “신조어 및 띄어쓰기 오류에 강인한 시퀀스-투-시퀀스 기반 한국어 형태소 분석기”. <정보과학회논문지> 47(1). 70-77. http://dx.doi.org/10.5626/JOK.2020.47.1.70

최석재. “RHINO”. https://github.com/SukjaeChoi/RHINO

최재혁, 이상조(1993). “양방향 최장일치법에 의한 한국어 형태소 분석기에서의 사전 검색 횟수 감소 방안”. <정보과학회논문지> 20(10). 1497-1507. https://www.riss.kr/link?id=A82292437

한경은, 백슬예, 임재수(2017). “공개와 협업을 통한 세종 형태 분석 말뭉치 오류 개선 방법”. <한국어정보학회 학술대회> 228-232. https://koreascience.kr/article/CFKO201712470015351.page

黃秀智(2023). 新聞を利用した日韓外来語の通時的対照研究. <동서인문학> 64. 115-140. https://www.riss.kr/link?id=A108495512

Aho, Alfred V; Corasick, Margaret J. (1975). “Efficient string matching: an aid to bibliographic search”. *Communications of the ACM*, *18*(6), 333-340. https://doi.org/10.1145/360825.360855

Chen, Stanley F; Goodman, Joshua. (1999). “An empirical study of smoothing techniques for language modeling.” *Computer Speech & Language*, 13(4), 359-394. https://doi.org/10.1006/csla.1999.0128

Devlin, Jacob; Chang, Ming-Wei; Lee, Kenton; Toutanova, Kristina. (2018). “Bert: Pre-training of deep bidirectional transformers for language understanding”. *arXiv preprint arXiv:1810.04805*. https://doi.org/10.48550/arXiv.1810.04805

Hofmann, Valentin; Schuetze, Hinrich; Pierrehumbert, Janet. (2022). “An Embarrassingly Simple Method to Mitigate Undesirable Properties of Pretrained Language Model Tokenizers.” *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics* (pp. 385–393). https://doi.org/10.18653/v1/2022.acl-short.43

KAIST Semantic Web Research Center. “HanNanum - SWRC”. https://web.archive.org/web/20111020055238/http://semanticweb.kaist.ac.kr/home/index.php/HanNanum

Kim, Ahrii and Kim, Jinhyeon. (2022, May). “Vacillating human correlation of sacrebleu in unprotected languages”. *Proceedings of the 2nd Workshop on Human Evaluation of NLP Systems (HumEval).* 1-15. https://doi.org/10.18653/v1/2022.humeval-1.1

Kim, Boseop; Kim, HyoungSeok; Lee, Sang-Woo; Lee, Gichang; Kwak, Donghyun; Jeon, Dong Hyeon; Park, Sunghyun; Kim, Sungju; Kim, Seonhoon; Seo, Dongpil. (2021, November). “What Changes Can Large-scale Language Models Bring? Intensive Study on HyperCLOVA: Billions-scale Korean Generative Pretrained Transformers.” *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing* (pp. 3405-3424). https://doi.org/10.18653/v1/2021.emnlp-main.274

Kim, Jae-Hoon; Jang, Byung-Gyu; Kim, Gil Chang; Seo, Jungyun. (1995). “Morphological ambiguity reduction using subsumption relation in Korean.” *Proceedings of the Natural Language Processing Pacific Rim Symposium (NLPRS’95)*.

Kim, Taewan; Bae, Seolyeong; Kim, Hyun Ah; Lee, Su-woo; Hong, Hwajung; Yang, Chanmo; Kim, Young-Ho. (2023). “MindfulDiary: Harnessing Large Language Model to Support Psychiatric Patients' Journaling”. *arXiv preprint arXiv:2310.05231*. https://doi.org/10.48550/arXiv.2310.05231

Kudo, Taku. “MeCab”. https://sourceforge.net/projects/mecab/

Kudo, Taku; Yamamoto, Kaoru; Matsumoto, Yuji. (2004, July). “Applying conditional random fields to Japanese morphological analysis”. *Proceedings of the 2004 conference on empirical methods in natural language processing* (pp. 230-237). https://aclanthology.org/W04-3230/

Lee, Do-Gil; Hae-Chang Rim. (2009). “Probabilistic modeling of Korean morphology.” *IEEE transactions on audio, speech, and language processing*, 17(5), 945-955. https://doi.org/10.1109/TASL.2009.2019922

Lee, Gary Geunbae; Cha, Jeongwon; Lee, Jong-Hyeok. (2002). “Syllable-pattern-based unknown-morpheme segmentation and estimation for hybrid part-of-speech tagging of Korean”. *Computational Linguistics*, 28(1), 53-70. https://doi.org/10.1162/089120102317341774

Lim, Heui-Suk; Lee, Sang-Zoo; Rim, Hae-Chang. (1995). “An efficient Korean morphological analysis using exclusive information.” *Proceedings of the 1995 International Conference on Computer Processing of Oriental Languages* (pp. 225-258).

Mikolov, Tomas; Sutskever, Ilya; Chen, Kai; Corrado, Greg S; Dean, Jeff. (2013). “Distributed representations of words and phrases and their compositionality”. *Advances in neural information processing systems*, 26. https://doi.org/10.48550/arXiv.1310.4546

open-korean-text. “open-korean-text”. https://github.com/open-korean-text/open-korean-text

Park, Kyubyong; Lee, Joohong; Jang, Seongbo; Jung, Dawoon. (2020, December). “An Empirical Study of Tokenization Strategies for Various Korean NLP Tasks.” *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing* (pp. 133-142). https://aclanthology.org/2020.aacl-main.17/