Subword 유닛과 역 번역을 이용한

한국어-영어 신경망 기계 번역

Korean-English Neural Machine Translation
Using Subword Unit and Back Translation

2018년

서강대학교 대학원

컴퓨터공학과

이 재 환

Subword 유닛과 역 번역을 이용한

한국어-영어 신경망 기계 번역

Korean-English Neural Machine Translation
Using Subword Unit and Back Translation

지도교수 서 정 연

이 논문을 공학석사 학위논문으로 제출함

2019년 1월

서강대학교 대학원

컴퓨터공학과

이 재 환

논 문 인 준 서

이재환의 공학석사 학위논문을 인준함

2019년 1월

주심 박 운 상 (인)

부심 서 정 연 (인)

부심 구 명 완 (인)

감사의 글

4년의 학부과정을 마치고 2년의 석사과정을 거쳐 이제야 서강을 떠나게 되었습니다. 그동안 감사했던 분들에게 이 글을 바칩니다.

우선 저를 제자로 받아주시고, 2 년간 학업에 있어서, 그리고 삶에 있어 많은 조언을 아끼지 않아 주신 서정연 교수님께 감사하다는 말씀을 드립니다. 돌이켜 생각해보면, 석사과정을 밟기 이전 학부과정 때에도 교수님을 많이 따랐던 것 같습니다. 그래서 입대 전, 그리고 전역 후에 제가 일부러 교수님께 상담을 요청드리기도 했었죠. 그때부터 교수님께서 해주셨던 말씀들이 제 인생의 방향을 잡는 데에 큰 도움이 되었던 것 같습니다. 감사합니다.

역시 마찬가지로 지난 2 년간 제 연구를 많이 도와주시고 다방면으로 많은 조언을 주셨던 고영중 교수님께도 감사하다는 말씀을 드리고 싶습니다. 서정연 교수님께서 석사로서의 저를 낳으셨다고 한다면 고영중 교수님께서는 함께 길러주신 분입니다. 그동안 정말 감사했습니다.

매주 저희 연구실 세미나에 참여하시며 역시 많은 조언을 아끼지 않으시고 제 졸업논문 심사를 맡아주신 구명완 교수님께도 감사하다는 말씀을 드리고 싶습니다.

제 졸업 논문 심사를 위해 흔쾌히 주심을 맡아주신 박운상 교수님께도 감사하다는 말씀을 드립니다.

학부 때에 우연한 계기로 연이 닿아, 석사 마지막 학기까지 많은 일을 도와드리게 되었던 이지선 교수님께도 감사하다는 말씀을 드립니다.

코스웍 중 머신러닝과 딥러닝 학습에 있어 많은 가르침을 주시고, 석사로서의

성실한 삶과 방향에 대해서 많은 고민을 하게 해주셨던 전자공학과 김경환 교수님께도 감사하다는 말씀을 드리고 싶습니다.

2 년간 매일 마주치며 함께 생활을 가장 많이 한 연구실 학우분들께도 감사하다는 말씀을 전하고 싶습니다.

우리 연구실의 맏형이자 믿음직스러운 박사님 영민이형, 같이 NMT 를 연구하며 정말 많은 도움을 주셔서 너무 감사한 광호형, 우리 연구실의 씩씩한 엄마이자 리더 주애, 옆방에서 함께 수많은 밤을 함께 지새우며 의지 아닌 의지가 되었던 윤석이형, 말이 필요없는 홍선이, 공부도 덕질도(?) 뭐든지 열심히 하는 유진이, 제일 오래 함께한 유일한 동기지만 함께 졸업 하진 못해 아쉬운 병재, 반년 뒤에 다시 만나게 될 사실상 동기 찬민이, 결혼 축하하고 축복하는 개그 코드가 제일 잘 맞았던 태석이, 학부 때부터 나름 쭉 같이 재미있게 지낼 수 있어서 좋았던 민영이, 뭔가 나와 비슷한 부분이 많은 것 같아 더 정이 갔던 휘진이, 정말 아는 사람이 많이 겹치는데 석사 때가 되어서야 직접적으로 알고 지내게 되어 아쉬운 영준이, 학부 때부터 친구였지만 석사를 하면서 더 잘 알고 지내게 되어 좋았던 보성이, 함께한 시간이 너무 짧아 매우 아쉬운 윤영이 모두 저와 함께 같은 기간 동안 연구실 생활을 해주셔서 감사합니다. 그리고 저희 연구실을 위해 늘 힘써 주셨던 안소영, 장리오, 정예준, 김진아 간사님께도 감사하다는 말씀을 드립니다.

친구들에게도 감사의 말을 전합니다. 이렇게나 좋은 사람들이 많다니 저는 정말 복 받은 사람입니다. 제일 먼저 학부 때부터 지금까지 늘 변함 없이 돈독한, 나의 믿을맨 3/4 친구들 상근 민승 정인에게 감사의 말을 전합니다.

고등학교때부터 10 년넘게 우정을 이어 나가고있는 마음의 고향같은 친구들 규현 재영 영규 윤석에게도 감사의 말을 전합니다.

다들 나만 쏙 빼고 먼저 사회로 나가 아쉽지만, 각자의 자리에서 빛나고 있는 자랑스러운 멋진 동기들 홍선 윤민 바켠 시넌 희문 승원 석현 상규 홍환 성필에게도 감사의 말을 전합니다.

학부 때부터 지금까지 쭉 제 삶에 큰 부분을 차지한 CSFC 형들, 동기들 그리고 동생들, 풋살 팀, 전주 군산 멤버 원섭 상현 지수 재성 홍선 + 도근에게 감사의 말을 전합니다. 앞으로도 쭉 즐겁게 공 찼으면 좋겠습니다.

역시 마찬가지로 학부 때부터 제 삶에 큰 부분을 차지한 ABYSS 12, 13, 14 기 형 동생 친구들, 벌레들 경민 원준 도현 재훈 경보 정우 세광 석준 우성에게 감사의 말을 전합니다. 다들 변함없이 늘 벌레였으면 좋겠습니다.

석사 때부터 들어가게 되어 지금 저의 삶에 정말 큰 부분을 차지하게 된 스트로크 형 동생 친구들, 소윤이를 중심으로 한 고중물 모임, 소윤 수민 홍선 경동 하균, 같은 공감대를 가지고 많은 고민들을 함께한 석사 친구들 대인 하연에게 감사의 말을 전합니다. 석사과정 중 스트로크에 들어간 것은 제가 석사 생활을 하며 가장 잘한 일 중 하나입니다,

역시 석사 때부터 시작한 제가 만든 밴드 JDS 소운 아현 석인 하균 영호 준형 재성에게도 감사의 말을 전합니다. 졸업 전 밴드 공연을 하고싶다는 제 욕심을 흔쾌히 수락해준 친구들에게 너무 고맙고, 단발성 밴드가 아니라 친해져서 많은 일상을 공유하고 있어 좋습니다. 꼭 언젠가 또 공연했으면 좋겠습니다.

매 달마다 적어도 하루씩을 기대하게 만들어주는 친구들 민승 나은 소정에게도 감사의 말을 전합니다. 앞으로도 맛있는 것들 많이 부수고 다녔으면 좋겠습니다.

학부 마지막 학기 졸업프로젝트 팀으로 시작해 꾸준히 교류를 이어나가고 있는 필리네어 친구들, 은아 연재 성필에게도 감사의 말을 전합니다.

미처 여기에 다 적지 못한 몇몇 친구들에게도 감사의 말을 전합니다. 만나서 꼭 보답하겠습니다.

가족들에게도 감사의 말을 전합니다. 저를 위해 기도해주시고 저를 응원해주시는 분들, 그리고 이제 전역 후의 삶을 열심히 살고 있는 동생에게 감사의 말을 전합니다. 그리고 부모님, 지금의 제가 있는 건 다 저를 올바르게 길러 주시고 제 선택과 결정을 늘 존중해주시고 항상 저를 믿어주고 지지해 주셨던 부모님 덕분입니다. 감사합니다 존경합니다 사랑합니다.

끝은 또다른 시작 이라고들 합니다. 앞으로도 열심히 살겠습니다 감사합니다.

2018년 12월 20일
서강대학교 R관 908호에서
이재환 올림

차 례

그림 차례

표 차례

Abstract

Neural Machine Translation is a technique that uses an artificial neural network to express an input sentence as a vector and output the translated sentence as an end-to-end sentence using the vector. And is showing a high translation performance.

In the early artificial neural network machine translation, a basic unit constituting a sentence was assumed to be a 'word', and a dictionary was constructed based on the word and used for translation. However, since the size of a dictionary used for translation can not be increased indefinitely, translation difficulties arise when words that have not been registered in advance appear (when the Out of Vocabulary problem appears). In order to solve this problem, a method of assuming a unit constituting a sentence as a subword unit has been proposed, and to be specific, a byte pair encoding.

Artificial neural network machine translation has high performance when there are large parallel corpus, but it does not perform very well for language pairs that do not have such large parallel corpus. To solve this problem, a method of increasing the number of corpus through a Back Translation technique has been proposed.

In this paper, we apply the BPE and Back Translation techniques in Korean - English artificial neural network machine translation. We can improve the performance of Korean - English artificial neural network machine translation by applying the two techniques.

Key words

Neural Machine Translation, Back Translation, Subword, Byte Pair Encoding, Machine Learning, Deep Learning, Sequence to Sequence, Encoder, Decoder

요 약

인공 신경망 기계번역(Neural Machine Translation)은 인공 신경망을 활용하여 입력 문장을 벡터로 표현하고 그 벡터를 이용하여 번역 문장을 End-to-End 방식으로 출력하는 기법으로, 최근 다른 통계적 기반 번역 방법에 비해 높은 번역 성능을 나타내고 있는 중이다.

초기의 인공 신경망 기계 번역은 문장을 구성하는 기본 단위를 '단어'로 상정하고 이를 기반으로 사전을 구성하여 번역에 이용 하였다. 하지만 번역에 사용되는 사전의 크기를 무한정 늘릴 수는 없으므로, 사전에 등록되지 않은 단어가 등장했을 때(Out of Vocabulary 문제가 등장했을 때) 번역이 어려운 문제가 발생한다. 이를 해결하기 위해, 문장을 구성하는 단위를 단어보다 더 세부적인 것(Subword Unit)으로 상정하는 방법이 제안되었고, 그 구체적인 방법으로는 Byte Pair Encoding 이 있다.

인공 신경망 기계번역은 대규모의 병렬 코퍼스가 존재할때에는 높은 성능을 나타내지만, 이런 대규모의 병렬 코퍼스가 존재하지 않는 언어쌍에 대해서는 그다지 높은 성능을 나타내지 못한다. 이를 해결하기 위해, Back Translation 기법을 통해 학습 코퍼스의 수를 증가시키는 방법이 제안된 바 있다.

본 논문에서는 한국어-영어 인공 신경망 기계 번역에 있어 BPE 기법과 Back Translation 기법을 적용하여 다양한 조건 하에 실험을 진행하였고, 두 기법을 적용시킴으로써, 한국어-영어 인공 신경망 기계 번역의 성능을 향상시킬 수 있음을 확인하였다.

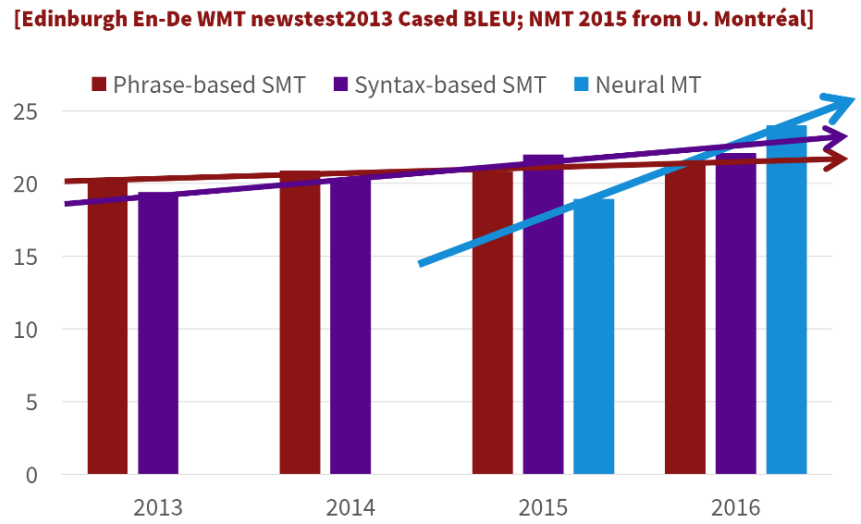
주제어 (색인어, 키워드)

기계번역, 역번역, 하위단어, 바이트쌍인코딩, 기계학습, 딥러닝, 시퀀스투시퀀스, 인코더, 디코더

1. 서 론

1.1. 연구의 배경 및 동기

인공 신경망 기계 번역은 end-to-end 접근법을 이용한 번역의 자동화를 가능하게 한 모델로서, 최근 일일이 수작업으로 해야하는 후처리가 필요하다는 단점을 가진 이전의 통계 기반 기계번역의 단점을 개선 시켜 나가며 번역 성능 또한 앞서나가고 있다.



[그림 1] 최근 SMT와 NMT의 WMT 영어-독일어 번역 성능 비교
출처 : http://www.meta-net.eu/events/meta-forum-2016/slides/09_sennrich.pdf

[그림 1]은 에딘버러 대학이 Workshop on Machine Translation(WMT) 이라고 하는 기계 번역 워크숍에 매년 자체적으로 참가하면서 내놓은 번역 모델의 성능

비교를 나타낸 그림이다. 그림에서 보듯이, 신경망 기반 기계 번역은 기존의 통계 기반의 번역 성능의 성장 속도를 가볍게 따라잡으면서 동시에 번역 성능의 절대치 또한 더 앞서고 있다. 현재 대부분의 기계번역 관련 연구는 end-to-end 방식의 신경망 기반 기계번역 분야가 연구되고 있다.

하지만 한국어-영어 언어 쌍에 대해서는 기존에 연구가 거의 진행되어 오지 않았다. 현재 대부분의 기계 번역 관련 연구에서는 영어, 독일어, 불어를 언어로 많이 채택하고 있다. 해당 언어들의 병렬 코퍼스가 많이 존재하기 때문이다. 한국어-영어 기계번역은 언어쌍의 병렬 코퍼스의 수가 많지 않은 일종의 low-resource 기계번역 task 이다.

1.2. 논문의 초점

본 연구에서는 한국어-영어 언어쌍의 기계 번역이 병렬 코퍼스의 수가 많지 않은 low-resource task 라는 점에 착안하여, low-resource 기계번역 task 에 적용되어 성능이 향상되었다는 연구 결과가 존재하는 back translation 이라는 방법을 적용하여, 학습에 사용되는 병렬 코퍼스의 양을 증대시키는 방법을 채택하여 연구를 진행해 보았다. 이와 더불어, 한국어가 단어 형성법이 풍부하고 복잡한 언어라는 점에 착안하여, 기존에 이와 같은 언어에 적용했을 때, 성능이 오른 바 있는, Subword unit 기법을 적용하여 연구를 진행해 보았다.

그 결과, Subword 기법을 적용하지 않고, 한국어와 영어 모두 단어를 단위로 하여 사전을 구축 했을 때 보다 Subword 기법인 Byte Pair Encoding 기법을 적용하여 사전을 구축 했을 때 번역 성능이 BLEU 4 기준으로 5.44 향상되었으며, back

translation 기법을 적용 한 뒤, 새롭게 생성된 영어 문장 중 번역 성능이 높은 문장만 선별하여 학습 코퍼스에 추가 시켰을 때의 성능은 새롭게 코퍼스를 추가시키지 않았을 때 보다 BLEU 4 기준으로 1.09 향상되었다.

1.3. 논문의 구성

본 논문의 구성은 다음과 같다. 2장에서는 본 논문과 관련이 있는 개념들과 관련 연구들을 소개한다. 3장에서는 본 논문에서 사용되는 중요한 개념인 Subword unit (Byte Pair Encoding)과 Back Translation 에 대해 설명하고, 4장에서는 실험과 관련하여 사용한 데이터, 실험에 사용한 학습 파라미터, 평가 방법, 그리고 실험 결과와 그에 대한 분석 결과를 소개한다. 끝으로 5장에서는 본 논문의 결론과 이와 관련하여 수행할 추후 연구에 대하여 기술한다.

2. 인공 신경망 기계 번역 관련 연구

본 장에서는 본 논문을 이해하는 것에 있어서 필요한 여러 가지 기본 개념에 대해 알아보고, 관련이 있는 기존 연구들에 관하여 알아본다. 기본 개념으로는 우선 기존의 번역에 활용하던 통계 기반 기계번역과 그로부터 발전된 인공 신경망 기계번역에 대해 다루고, 본 논문에서 다루는 subword 유닛 기법과 역 번역 기법에 대해 다룬다.

2.1. 통계 기반 기계번역(Statistical Machine Translation)

통계 기반의 기계번역은 통계학적 번역 모델을 이용한다. 통계학적 번역 모델들은 대량의 병렬 코퍼스를 분석하여 얻은 정보를 바탕으로 구축된다. 즉, 해당 언어의 구조 및 특성에 대해 자세히 알지 못한다 할지라도, 병렬 코퍼스에 나타난 대상 언어쌍의 언어적 특성을 통계적으로 분석, 모델링 하여 번역에 필요한 지식을 자동으로 추출하고 번역할 수 있게 한다. 통계학적 번역 모델의 구축은 매우 빠르게 진행될 수 있지만, 다국어 병렬 코퍼스의 수가 매우 많아야 하며, 이 병렬 코퍼스에 매우 의존적이라는 단점이 있다[1].

통계학적 번역 모델의 예를 들자면, Source언어가 불어(f)이고 target언어가 영어€라고 가정할 때, 베이즈의 법칙을 이용하여 다음과 같은 확률식을 세운다.

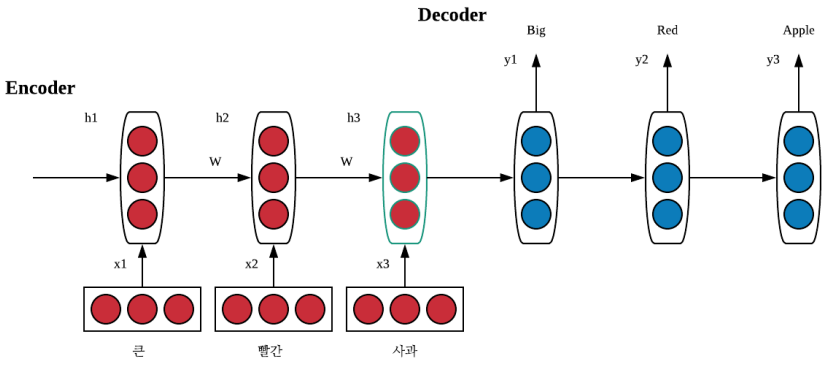
$$\hat{e} = argmax_e p(e|f) = argmax_e p(f|e)p(e)$$

번역 모델 $p(f|e)$는 불어-영어 병렬 코퍼스로부터 특정 불어 source 문장이 어떻게 target 영어 문장으로 번역될 수 있는지 학습되는 부분이고, $p(e)$는 영어 단일 코퍼스

로부터 학습되어, 영어 문장으로의 번역 과정에서 문법적으로나 표현상 더 자연스러운 문장 생성이 가능하도록 해주는 language model이다.

번역 모델의 학습을 위해선 우선 특정 불어 표현이 특정 영어 표현과 의미가 상응되도록 정렬 과정을 거쳐야 한다. 그러나 언어의 특성상, 특정 불어 표현은 영어에서 사용이 되지 않기도 하고, 하나의 불어 단어가 영어에선 여러 단어로 표현이 되기도, 반대로 여러 불어 어구들이 모여 하나의 영어 단어의 뜻을 나타내기도 하며, 여러 불어 어구들이 여러 영어 어구와 동일한 뜻을 나타내기도 한다. 이렇게 어렵고 복잡한 정렬 과정을 거친 후에는 여러 번역 가능한 경우 수 중 한가지를 선택해야 하는데, 이것 또한 상당히 어려운 탐색 문제이며, 여기서도 language model이 포함되기도 한다. 이렇듯, 통계 기반 기계번역은 그 과정이 매우 복잡하고, 한번에 학습이 불가능하고 여러 부분을 따로따로 학습해야 하며, 별도의 많은 feature engineering이 필요하다.

2.2. 인공 신경망 기계번역(Neural Machine Translation)



[그림 2] RNN Encoder Decoder를 이용한 seq2seq 인공 신경망 기계번역

[그림 2]는 통계 기반 번역의 문제점을 해결하고자 등장한 RNN 인코더 디코더를 이용한 sequence to sequence 인공 신경망 기계 번역의 전체적인 구조를 나타낸다. End to end로 학습되어 별다른 복잡한 후처리나 feature engineering이 필요하지 않아 용이하며, 현재 활발한 연구를 통해 통계 기반 번역의 성능을 뛰어넘는 성능을 보여주고 있다.

인코더에서 진행되는 기작을 수식으로 나타내면 다음과 같다.

$$h_t = \mathrm{f}(W^{(hh)}h_{t-1} + W^{(hx)}x_t)$$

여기서 W는 가중치를 나타내며, ht는 hidden state를 나타내며, xt는 현재 시점의 input의 word vector를 의미한다. 즉, 현재 시점의 hidden state는 이전 시점의 hidden state에 가중치를 곱한 것과 현재 input vector에 가중치를 곱한 것의 합으로 나타낼 수 있다.

디코더에서 진행되는 기작을 수식으로 나타내면 다음과 같다.

$$h_t = f(W^{(hh)}h_{t-1})$$

$$y_t = softmax(W^{(S)}h_t)$$

디코딩 과정에서는 input이 없기 때문에 단순히 현재 시점의 hidden state는 이전 시점의 hidden state에 가중치를 곱한 것이며, 이 hidden state에 가중치를 곱해서 softmax를 취해 가장 높은 값으로 출력되는 것이 현재 시점의 output, y_t임을 알 수 있다.

전체 모델은 다음의 cross entropy function을 최소로 만들도록 학습된다.

$$\max_{\theta} \frac{1}{N}\sum_{n=1}^{N} log\, p_\theta(y^{(n)}|x^{(n)})$$

즉, 특정 input x 에 대해 output y 가 가장 유사하도록 학습을 진행한다.

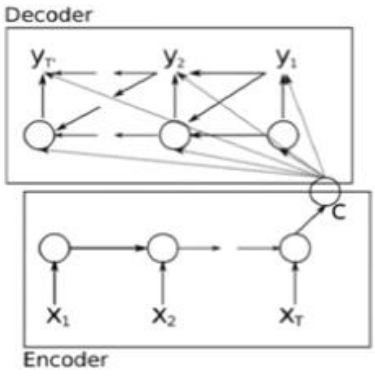
그러나 현실적으로 이 모델은 너무 성능이 낮기 때문에 몇가지 발전된 방법이 기존

의 모델에 추가된다. 첫째로는 디코딩 과정에서 매 time step마다 Encoder의 마지막 hidden vector인 C와 이전 출력인 y_{t-1}을 넣어준다.

$\emptyset$함수를 가중치 W와의 곱이라고 할 때, (ex : $h_t = \emptyset(h_{t-1}) = \mathrm{f}(W^{(hh)}h_{t-1})$) 이는 다음의 수식처럼 표현할 수 있다.

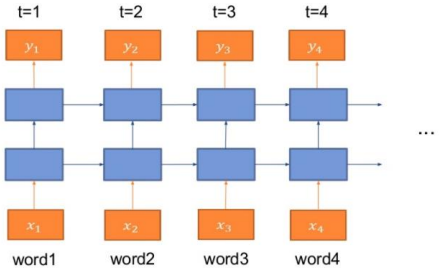
$$h_{D,t} = \emptyset_D(h_{t-1}, c, y_{t-1})$$

또한 이를 그림으로 나타내면 아래의 [그림 3] 과 같다[3].



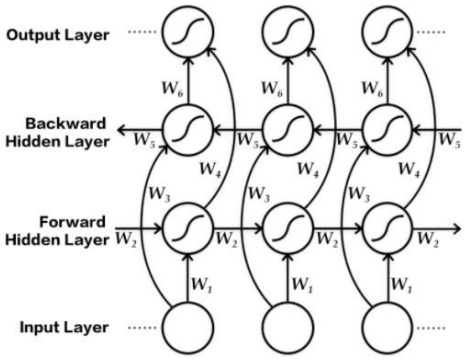
[그림 3] 디코딩 과정에 C와 y_{t-1}를 추가한 모델

두번째로는 여러 층을 쌓은 RNN모델을 학습시키기도 한다. 이를 그림으로 나타내면 아래의 [그림 4] 와 같다[4].



[그림 4] 여러층이 쌓인 RNN 모델

세번째로는 RNN을 겹겹이 쌓는것에 더하여 정방향과 역방향으로, 양방향으로 진행하는 encoder를 학습한다. 이를 그림으로 나타내면 아래의 [그림 5] 와 같다[5].



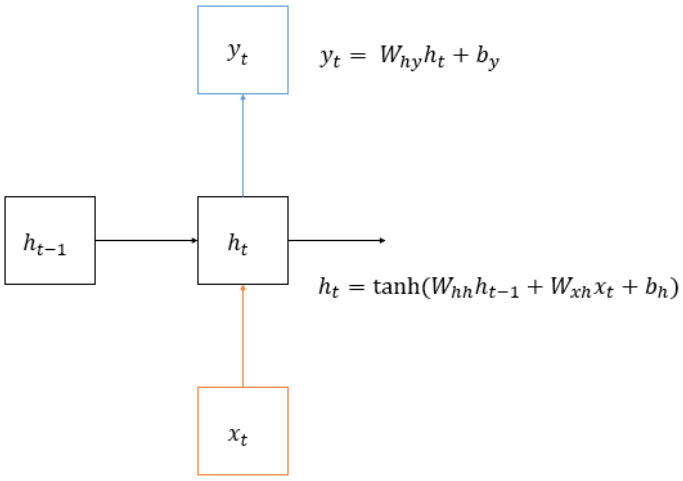
[그림 5] 양방향 encoder

마지막으로는 input으로 들어가는 단어들의 순서를 거꾸로 뒤집어서 input으로 넣어주는 방법이 있다. 즉, 원래 A B C가 input으로 들어가서 output으로 X Y가 출력되는 모델이었다면, input의 순서를 거꾸로 뒤집어서 C B A -> X Y가 되도록 모델을 학습하는 방법이다. 이는 입력 단어와 출력 단어 사이의 거리를 가깝게 해주는 역할을 하여, 역 전파 과정에서 단어가 덜 잊혀져 더 정확한 output이 나오도록 하는 데에 도움을 준다[2].

RNN cell로 어떤 것을 사용하는지에 따라 성능이 달라지기도 한다. RNN cell에는 주로 Long Short Term Memories(LSTM) 혹은 Gated Recurrent Unit(GRU)를 사용한다.

2.3. Recurrent Neural Networks(RNNs)

Recurrent Neural Network는 한 노드의 hidden state 값이 다시 반복적으로 다음 노드에 연결되어 input으로 들어가는, 순환 구조를 이루는 인공 신경망의 한 종류이다. RNN은 시퀀스 길이에 관계 없이 input을 받아 output을 낼 수 있는 네트워크 구조이기 때문에, 필요에 따라 다양한 task에 있어 유연하게 구조를 만들 수 있다는 장점이 있다.



[그림 6] Recurrent Neural Network의 구조

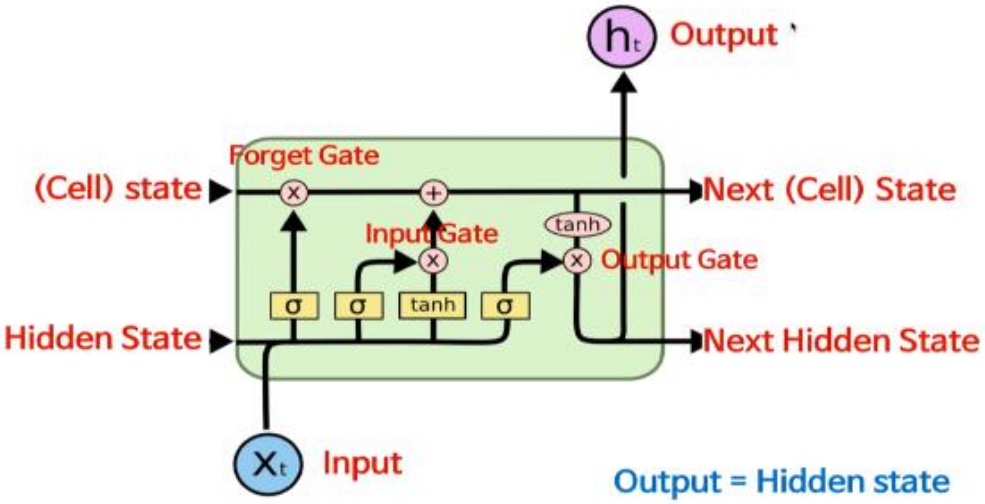
위의 [그림 6]은 RNN cell의 구조를 나타낸다. x_t는 현재시간 t에서의 input을 나타내며, y_t는 현재시간 t에서의 output을 나타낸다. 현재시간 t에서의 상태 h_t는 직전 시점의 hidden state h_{t-1}와 x_t가 합쳐져 갱신된다.

RNN은 서로 시간적으로 correlation이 있는 sequencial한 데이터를 처리하기 위해 사용되는데, 시간이 많이 지나면 오래된 정보들은 빨리 잊어버리게 되는 경향이

있어, 학습 능력이 저하된다는 단점이 있다. 이를 vanishing gradient problem이라고 한다. 그리고 이 문제를 해결하기 위해 고안된 것이 바로 Long Short Term Memory cell이다.

2.4. Long Short Term Memory(LSTM)

LSTM은 RNN의 hidden state에 cell의 state를 추가한 구조를 나타낸 것으로, 구체적으로 시간의 흐름에 따른 정보를 유지, 삭제, 추가하는 역할을 하는 3개의 비선형 게이트(non-linear gate)와 이들 게이트를 통해 정보를 유지하는 Memory cell(Cell State)의 도입을 통해 오래전의 정보도 더 오래 기억할 수 있도록 한다[6].



[그림 7] Long Short Term Memory의 구조
출처 : http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[그림 7]은 LSTM cell의 구조를 나타낸다. LSTM의 핵심은 Cell state이다. Cell state는 지금까지 가지고있는 정보를 기반으로 취합된 새로운 정보가 흘러가는 흐름을 나타낸다. 이러한 정보의 취합 혹은 통제는 3개의 게이트를 통해 이루어진다.

첫째는 forget gate이며, 다음과 같은 수식으로 표현할 수 있다.

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right)$$

Forget gate는 이전 hidden state와 현재 input을 입력으로 받은 뒤, sigmoid를 통해 0~1사이의 값으로 나오는 결과값 f_t으로, cell state에 존재하는 어떤 성분값을 제거할지, 혹은 남길지를 결정하는 gate이다.

두번째는 input gate이며, 다음과 같은 수식으로 나타낼 수 있다.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C$$

Input gate는 이전 hidden state와 현재 input을 입력으로 받은 뒤, tanh를 통해 -1~1사이의 값으로 나오는 일종의 cell state candidate값 $\tilde{C}_t$와, 마찬가지로 이전 hidden state와 현재 input을 입력으로 받은 뒤, sigmoid를 통해 0~1사이의 값으로 나오는 결과값i_t 의 element wise 곱 값 $i_t * \tilde{C}_t$를 기존의 cell state에 얼만큼 더해줄지 말지를 결정하는 gate이다.

Cell state를 업데이트 하는 과정은 forget gate와 input gate를 통해 진행되게 되는데 다음의 수식에 따른다.

$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

즉, 이전 cell state에 존재하는 값을 얼마나 제거할지, 새로운 input을 얼마나 더해줄지를 반영하여 cell state를 update시켜준다.

세번째는 output gate이며, 다음과 같은 수식으로 나타낼 수 있다.

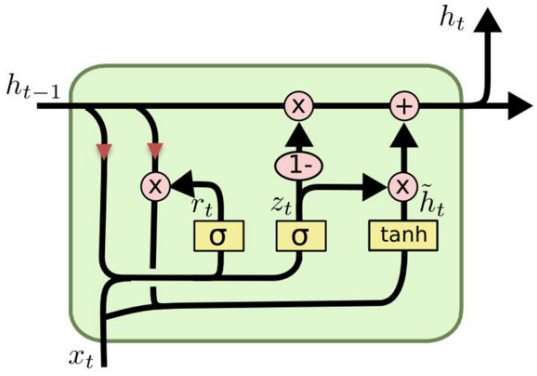
$$O_t = \sigma(W_O[h_{t-1}, x_t] + b_O)$$

$$h_t = O_t * \tanh(C_t)$$

Output gate는 update과정을 통해 얻어진 최종 cell state 값에 tanh를 적용하여 −1~1사이의 값으로 나오는 $\tanh(C_t)$값과, 이전 hidden state와 현재 input을 입력으로 받은 뒤, sigmoid를 통해 0~1사이의 값으로 나오는 결과값 O_t의 element wise 곱 $O_t * \tanh(C_t)$, 즉 새로운 hidden state 값 h_t를 얼마나 다음 RNN cell로 넘겨줄지, 얼마나 밖으로 빼낼지를 정하는 gate이다.

2.5. Gated Recurrent Unit(GRU)

Gated Recurrent Unit은 기억을 오래 유지할 수 있다는 LSTM의 장점을 유지하면서 일부 게이트를 제외시켜, 계산 복잡성을 낮춘 cell 구조이다. 성능은 LSTM과 유사한 것으로 알려져 있다. GRU에는 reset gate와 update gate 두가지 게이트 만이 존재한다[7].



[그림 8] Gated Recurrent Unit의 구조
출처 : http://colah.github.io/posts/2015-08-Understanding-LSTMs/

[그림 8]은 Gated Recurrent Unit의 구조를 나타낸다. Reset gate는 다음과 같은 수식으로 나타낼 수 있다.

$$r_t = \sigma(W^{(r)}x_t + U^{(z)}h_{t-1})$$

$$\tilde{h}_t = \tanh(Wx_t + r_t \odot Uh_{t-1})$$

Reset gate는 현 시점에서 기억해 둘 만한 정보를 $\tilde{h}_t$만큼 정하는데, 현시점 정보(Wx_t)와 과거 정보(Uh_{t-1})를 반영하되, 과거 정보를 얼마나 반영할지는 이전 hidden state와 현재 input을 입력으로 받은 뒤, sigmoid를 통해 0~1사이의 값으로 나오는 결과값 r_t를 따른다.

Update gate는 다음과 같은 수식으로 나타낼 수 있다.

$$z_t = \sigma(W^{(z)}x_t + U^{(z)}h_{t-1})$$

$$h_t = z_t \odot h_{t-1} + (1 - z_t) \odot \tilde{h}_t$$

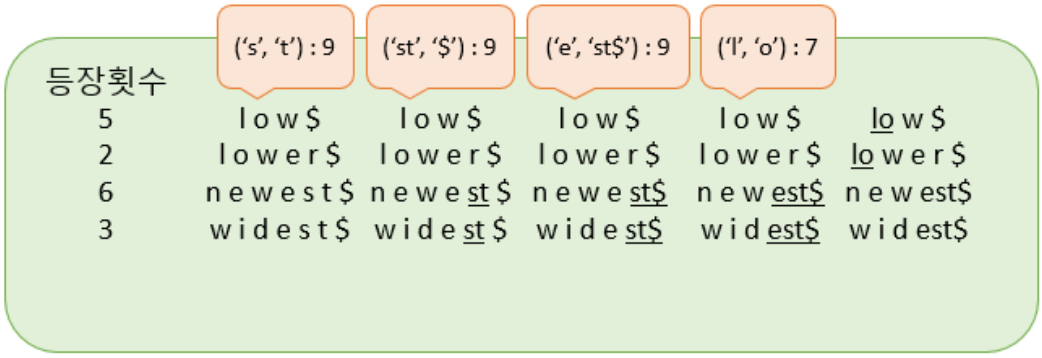
Update gate는 과거의 정보 h_{t-1}과 위의 reset gate를 통해 정한 현시점에 기억해 둘만한 정보 $\tilde{h}_t$를 적절히 조합하여 현재 시점의 정보를 update하는 역할을 하는데, 이를 얼마만큼씩 반영할지는 이전 hidden state와 현재 input을 입력으로 받은 뒤, sigmoid를 통해 0~1사이의 값으로 나오는 결과값 z_t를 따른다.

3. Subword 유닛 기법과 역 번역 기법

3.1. Byte Pair Encoding(BPE)

기존의 인공 신경망 기계번역은 문장을 구성하는 기본 단위를 '단어'로 상정하고 이를 기반으로 사전을 구성하여 번역에 이용했지만, 사전의 크기를 무한정 늘릴 수 는 없으므로 사전에 등록되지 않은 단어가 입/출력 문장에 출현 했을 때(Out of Vocabulary가 등장했을 때), 번역이 어려운 문제가 발생했다. 이를 해결하기 위해 문장을 구성하는 기본 단위를 단어보다 더 세부적인 것(Subword unit)으로 상정하는 방법이 제안되었고, 그 대표적인 방법으로 Byte Pair Encoding(BPE) 방법이 있다.[8].

이 방법을 사용하면 문장을 구성하는 기본 단위를 '단어'로 상정하고 이를 기반으로 구성한 사전의 크기보다 훨씬 더 작은 크기의 사전을 이용하여 학습 말뭉치의 모든 단어들을 다룰 수 있다는 장점이 있으며, 말뭉치에서 자주 등장하지 않아 사전에 등록되지 않은 개체명이나 고유명사, 숫자, URL주소 와 같은 요소의 번역을 효과적으로 해낼 수 있음이 입증되어 있다.



[그림 9] Byte Pair Encoding 기법의 동작 예

Byte Pair Encoding은 언어 단위 처리 체계에서 Subword를 기본으로 하는 Segmentation 기법으로, 특정 언어에 종속적인 문법적, 의미적 규칙이 필요하지 않고 오로지 데이터에만 의존적인 학습 방법이다. BPE 방법은 다음과 같은 알고리즘을 통해 진행된다.

1. 먼저 단어를 캐릭터 단위로 다 분할을 한 뒤, 전체 코퍼스에서 등장한 횟수를 기록한다.

2. 등장한 횟수가 가장 많은 공통의 캐릭터 bigram을 시작으로 병합하며, 더 이상 공통의 캐릭터 bigram이 존재하지 않을 때까지 BPE 기법을 적용하여 Subword 사전을 저장시켜 나간다.

3. Test time에는 새로 입력된 문장을 마찬가지로 캐릭터 단위로 다 분할을 한 뒤, 사전에 저장되어 있는 Subword를 적용시켜 입력 문장을 Subword 기반으로 분해한다.

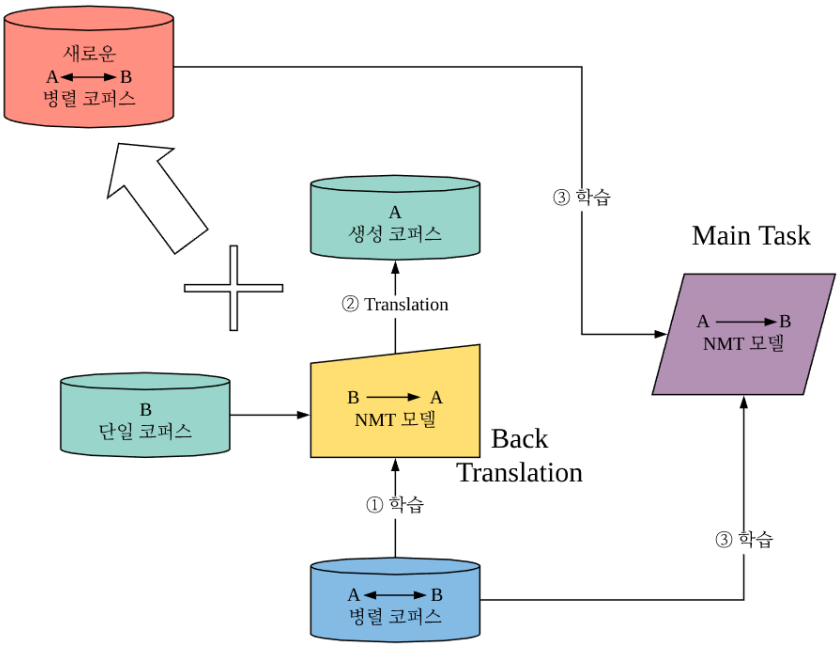
[그림 9]는 Byte Pair Encoding 기법이 어떻게 동작하는지를 그 예시를 나타낸다. 코퍼스에 단어 'low'가 5번, 'lower'가 2번, 'newest'가 6번, 'widest'가 3번 등장했고, 각각을 먼저 캐릭터 단위로 다 분할을 한다. 그 다음 가장 등장한 횟수가 많은 공통

의 캐릭터 bigram을 합쳐 나가기 시작하는데, 맨 처음 등장 횟수가 많은 bigram은 ('s', 't')라서 이 둘을 먼저 합쳤고, 그 다음은 ('st', '$'), ('e', 'st$'), ('l', 'o') 순으로 합쳐 나간다. 영어를 source 언어로 하고, 한국어를 target 언어로 했을 때, 모델의 변형 없이 영어의 기본 번역 단위를 단어로, 한국어의 번역 단위를 형태소로 했을 때의 모델보다 영어와 한국어 모두에 BPE를 적용시켜서 번역 했을 때, 성능향상이 됨이 입증되어 있다[9].

3.2. 역 번역(Back Translation)

기계번역은 양이 충분한 병렬 코퍼스에 대한 학습을 통해서는 좋은 성능을 나타냈지만, low resource, 즉, 그 양이 충분하지 못한 언어 쌍의 병렬 코퍼스에 대해서는 좋지 못한 성능을 나타낸다. 그래서 target side의 monolingual 데이터를 활용하여 원래 목표했던 번역 방향의 역방향으로 번역을 하여, 즉, Back Translation(역 번역) 과정을 통해 병렬 코퍼스를 증대시키는 방법이 연구된 바 있다[10]. 구체적인 방법으로는 목표했던 번역 방향의 역방향으로 번역을 할 때, 목표했던 번역방향으로 번역을 할 때 사용하는 target side의 monolingual data를 input으로 사용하고, output이 나오면 해당 input과 output을 한 쌍의 병렬 코퍼스로 간주하여 원래 학습 코퍼스에 추가한다.

아래 [그림 10]은 역 번역 기법이 전체적으로 어떻게 동작 하는지를 나타낸다.



[그림 10] Back Translation 기법의 동작 방법

먼저, 전체 시스템의 Main task 는 A 언어에서 B 언어로의 NMT 모델을 구축하는 것이라 정의한다. 첫째로, 기존에 있던 A 언어와 B 언어 쌍의 병렬코퍼스를 이용하여 B 언어에서 A 언어로의 sub task 기계번역모델을 학습시킨다. 둘째로, Main task 의 target 언어인 B 언어 단일 코퍼스를, 앞서 학습시킨 B 언어에서 A 언어로의 기계번역모델의 input 으로 넣어서 A 언어로 된 새로운 단일 코퍼스를 얻는다. 그리고 이 둘을 합쳐서 새로운 병렬코퍼스를 얻어내고, 기존의 병렬코퍼스와 이 새로 얻어낸 병렬 코퍼스를 합쳐서, 전체 시스템의 main task 인 A 언어에서 B 언어로의 NMT 모델을 구축 하는 데에 필요한 학습 코퍼스로 사용한다.

Main task 의 target 언어인 B 언어로 된 단일 코퍼스를 역 번역의 input 으로 사용하는 이유는, main task 를 위해 새로 얻어진 병렬 코퍼스에서 정답이 정확한

것이 더 유창한 단어 표현을 가능하게 해주어, 더 번역 성능을 높여주기 때문이다. 즉, 새로 얻어진 병렬코퍼스는 (B 언어 기존 단일 코퍼스, A 언어 생성 코퍼스) 가 되는데, main task 에서는 input 으로 A 언어 생성 코퍼스, 정답으로 B 언어 기존 단일 코퍼스를 사용하므로, 번역 성능이 올라간다. 만약 코퍼스 확장을 위해 Source 언어인 A 언어로 된 단일 코퍼스를 sub task 의 input 으로 사용한다면, 그 output 으로는 B 언어 생성코퍼스가 얻어지게 되는데, main task 인 A 언어에서 B 언어로의 번역에서 input 으로 A 언어 단일 코퍼스, 정답으로 B 언어 생성코퍼스를 사용하게 되면, B 언어 생성코퍼스가 완벽하게 올바른 문장이 아닐 확률이 높으므로, 번역 성능이 오히려 떨어질 수 있다.

4. 실험 및 결과 분석

본 장 에서는 실제 한국어-영어 기계번역에 있어, 어떠한 모델을 사용했는지에 대한 설명을 하고, 실험 데이터는 어떤 것을 사용했는지, 실험 환경은 어떠했는지를 나타낸다. 번역의 성능은 BLEU 라는 metric 을 사용하는데, 이에 대한 간단한 설명과 더불어서 실제 한국어-영어 기계번역에 BPE 기법과 역 번역 기법을 적용 했을 때, BLEU 성능이 어느정도 나오는지 나타내고, 결과를 분석한다.

4.1. 실험 구성

4.1.1. 실험에 사용된 데이터

실험에 사용한 데이터는 한국어-영어 병렬코퍼스인 OpenSubtitles2018 (http://opus.nlpl.eu/OpenSubtitles2018.php)과 영어 단일 코퍼스 WMT16 (http://www.statmt.org/wmt16/translation-task.html#download) 데이터이다.

OpenSubtitles2018 병렬 코퍼스의 경우에는 3 단어 이상으로 구성된 문장을 추출했으며, WMT16 데이터의 경우에는 3 단어 이상 15 단어 이하로 구성된 문장을 추출했다. [표 1]은 실험에 사용된 데이터의 통계들이다.

[표 1] 실험에 사용된 코퍼스 통계

OpenSubtitles2018 (EN-Ko Parallel)	Training Set	Dev Set	Test Set
	917,570	5,000	5,000
WMT16 (En Monolingual)	Dataset		
	982, 807		

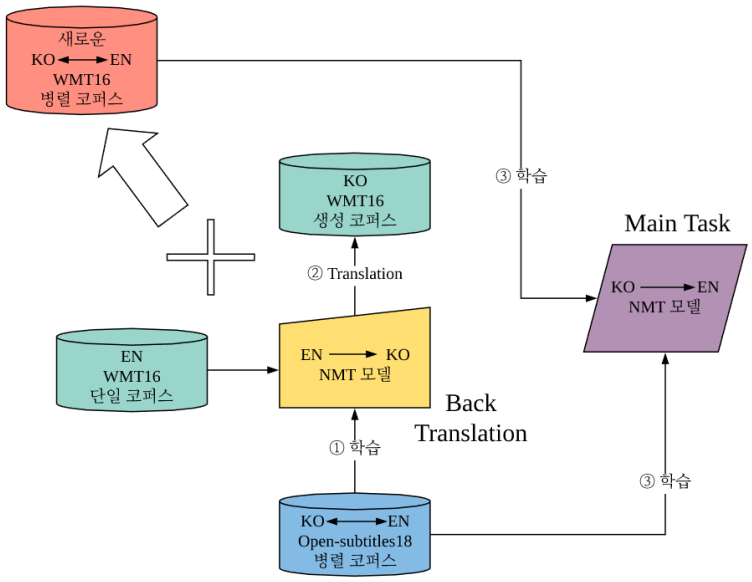
4.1.2. 실험 환경(Hyperparameter)

[표 2] 실험에 사용된 hyperparameters 상세 값

Attention	Scaled Luong	Num Train Steps	400000
Batch Size	128	Num Units	256
Dropout	0.2	Optimizer	Adam
Encoder Type	Bi-Directional	Unit Type	LSTM
Initial Weight	0.1	Beam Width	10
Learning Rate	0.0001	BPE Merge Operations	16000 / 32000
Num Layers	4	Data Tokenizer	Mosesdecoder

[표 2]는 실험에 사용된 하이퍼 파라미터(Hyperparameter)들의 상세 값을 나타낸다. 몇가지에 대해 설명하자면 Attention 은 Scaled Luong attention[11] 을 사용하였고, Optimizer 로는 Adam[12]를 사용하였으며, BPE 의 Merge Operation 은 16000 회와 32000 회를 사용했다. BPE Merge 횟수에 따른 한국어-영어 기계번역 성능을 비교해보고자 횟수를 다르게 하였고, 100 만문장 이하의 병렬코퍼스에 대해선 merge 16000 회를, 나머지 일반적인 상황에서는 merge 32000 회를 적용하는 것이 좋다는 연구결과가 있다[13]. 또한 모든 데이터들은 moses decoder 를 이용하여 tokenize 되고 정제되었다[14].

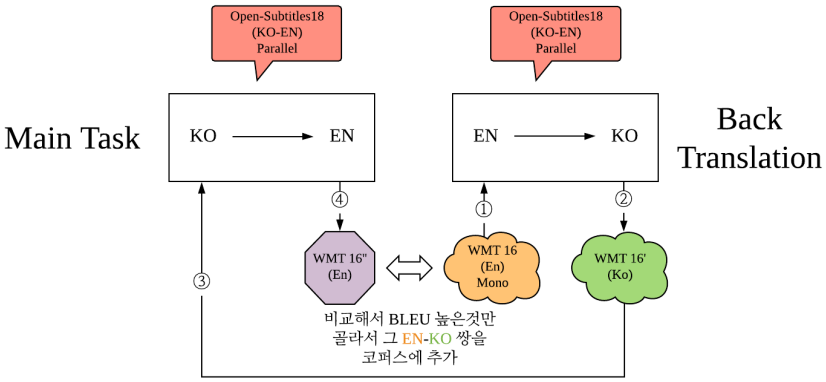
4.2. 제안 모델



[그림 11] Back Translation을 활용한 제안 모델

본 논문에서는 Subword unit 기법 중 하나인 Byte Pair Encoding 기법과 Back Translation 기법의 효과를 보이고자 하므로 [그림 11]과 같은 방식을 통해 실험 코퍼스에 Back Translation 기법을 적용하여 코퍼스를 확장시키고, Source 언어와 Target 언어 모두에 BPE 기법을 적용하였다.

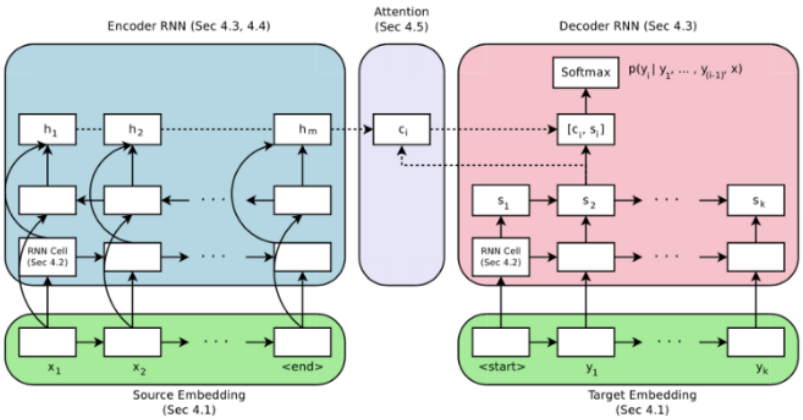
Back Translation 기법을 적용하여 코퍼스를 확장 시킬 때, 모든 코퍼스를 전부 Main Task 를 위한 새로운 코퍼스로 포함시키는 방법을 이용하여 실험을 한번 진행하였고, BLEU 점수를 이용하여, Back Translation 성능이 높은 생성 코퍼스만 선별하여 새로운 코퍼스로 포함시키는 방법을 이용하여 실험을 진행해보기도 했다.



[그림 12] BLEU 점수가 높은 생성 코퍼스만 선별하여 사용하기 위한 방법

위 [그림 12]는 Back Translation 성능이 높은 생성 코퍼스만 선별하여 새로운 코퍼스로 포함시키기 위한 방법을 나타낸다.

번역 모델은 구글이 https://github.com/tensorflow/nmt 에 공개한 기본적인 Sequence to Sequence, Encoder-Decoder 모델을 사용하였다. 전체적인 모델의 구조는 아래 [그림 13] 와 같다[15].



[그림 13] Google Sequence to Sequence, Encoder-Decoder 모델

4.3. 평가 방법

실험에 사용된 평가방법으로는 BLEU 점수를 통한 비교를 채택하였다[16]. BLEU 점수란, Source 문장에 대한 사람의 번역 결과와 번역 모델의 번역 결과가 얼마나 유사한지를 수치화 한 것으로, 수치화 하는 방식은 문장을 구성하는 단어들을 n-gram 으로 만든 뒤, 그 n-gram 들이 얼마나 서로 매칭되는지를 통해 판별한다. 1-gram, 2-gram, 3-gram, 4-gram 성능을 나타내는 BLEU1, BLEU2, BLEU3, BLEU4 점수를 통해 성능을 평가하였다. 보통의 BLEU 는 1 을 n-gram 이 100% 일치할 때의 성능으로 기준을 삼고 표기하지만, 편의에 따라 소수점을 최대한 없애기 위해 x100 을 하여 표기하기도 한다. 예를 들어 0.223 을 22.3 으로 표기하는 식이다. 본 논문에서는 후자의 표기방식을 사용한다.

4.4. 실험 결과 및 분석

본 장에서는 앞에서 제시한 실험 환경과 제안 모델, 평가 방법을 활용하여 본 연구를 통해 구축한 한국어-영어 기계번역 모델의 성능을 제시하고, 이를 분석한다. 먼저 BPE 를 적용시키지 않은 Baseline 모델과 BPE 를 적용시킨 모델의 성능을 비교하고, 그 다음으로는 BPE 를 적용시킨 기존 병렬코퍼스만 사용한 번역 모델의 성능과 기존 병렬 코퍼스에 새로운 역 번역 데이터를 추가한 번역 모델의 성능을 비교한다. 마지막으로는 BPE 를 적용시킨 기존 병렬 코퍼스만 사용한 번역 모델의 성능과 기존 병렬 코퍼스에 선택적으로 번역 성능이 높게 나온 새로운 역 번역 데이터를 추가한 번역 모델의 성능을 비교한다.

4.4.1 Baseline 모델과 BPE를 적용시킨 모델의 성능 비교

Baseline 모델로는 OpenSubtitles2018 코퍼스를 사용하였고, Source 언어인 한글과 Target 언어인 영어 모두 Word 단위로 나눈 모델을 사용한다. 사전 크기는 한글과 영어 모두 각각 50,000 단어이다. BPE 모델은 한국어 Source 문장을 형태소 분석 한 뒤 BPE 를 적용하였고, 영어 Target 문장에도 BPE 를 적용하였다. 앞서 서술했듯, BPE 는 merge 16000 회와 32000 회를 각각 적용시켜 실험했고, merge 16000 회 일 때의 사전 크기는 한국어, 영어를 섞어서 20,394 단어이며, merge 32000 회 일 때의 사전 크기는 한국어, 영어를 섞어서 35,970 단어이다. 또한 전체 코퍼스를 20 만, 40 만, 60 만, 80 만, 전체로 나누어 사용하여 실험하였다. 그 결과는 아래 [표 3], [표 4], [표 5] 와 같다.

[표 3] Baseline(Ko-Word, En-Word) 모델의 성능

	BLEU1	BLEU2	BLEU3	BLEU4
0.2m	25.07	14.28	9.33	6.49
0.4m	27.80	17.69	12.43	9.14
0.6m	30.77	20.25	14.57	10.99
0.8m	31.69	21.19	15.41	11.71
all	33.08	19.96	16.33	12.49

[표 4] BPE merge 16000(Ko-BPE, En-BPE) 모델의 성능

	BLEU1	BLEU2	BLEU3	BLEU4
0.2m	28.35	19.62	15.21	11.82
0.4m	33.25	24.32	19.40	15.74
0.6m	35.01	25.90	20.79	16.97
0.8m	35.55	26.46	21.38	17.14
all	35.4	26.38	21.27	17.44

[표 5] BPE merge 32000(Ko-BPE, En-BPE) 모델의 성능

	BLEU1	BLEU2	BLEU3	BLEU4
0.2m	27.43	18.53	14.12	10.78
0.4m	31.51	22.93	18.28	14.75
0.6m	33.83	24.90	19.94	16.18
0.8m	35.06	25.92	20.74	17.15
all	36.51	27.38	21.91	17.93

실험 결과, 단순히 BPE 를 Source 언어인 한글과 Target 언어인 영어에 적용시키는 것 만으로도 상당한 성능 향상을 얻어낼 수 있음을 확인하였으며, 당연하게도 코퍼스 수가 증가할수록 번역 성능이 높아짐을 확인 하였다. 그리고, [13]에서는 100 만문장 이하의 코퍼스에 대해서는 BPE merge 16000 의 성능이 BPE merge 32000 의 성능보다 더 좋다고 하였는데, 실험 결과 한국어-영어 기계 번역에서는 약 80 만 문장 정도를 경계로 하여 그 이하의 코퍼스 수에서는 BPE merge 16000 의 성능이 더 좋음이 확인 되었고, 그 이상을 넘어갈 때에는 BPE merge 32000 의 성능이 더 좋음을 확인 하였다.

4.4.2 기존 병렬 코퍼스만 사용한 번역 모델과 역 번역 데이터를 추가한 번역 모델의 성능 비교

역 번역 데이터를 추가할 때에는 WMT16 영어 단일 코퍼스 약 98 만문장중에서 45 만문장을 추출하여 이에 BPE merge 16000 과 BPE merge 32000 을 각각 적용시킨 실험을 진행하였고, 전체 98 만문장에 BPE merge 32000 을 적용시킨 실험을 진행하였다. 사전의 크기는 위의 실험과 거의 유사하다. 아래의 [표 6], [표 7], [표 8]은 각각의 번역 성능을 나타낸다.

[표 6] 기존 병렬 코퍼스 + 새로운 역 번역 데이터 0.45m BPE merge 16000의 성능

	BLEU1	BLEU2	BLEU3	BLEU4
0.2m	29.35	20.10	15.47	11.88
0.4m	31.91	23.17	18.51	14.99
0.6m	34.31	25.23	20.22	16.48
0.8m	34.39	25.52	20.62	16.86
all	35.43	26.30	21.12	17.19

[표 7] 기존 병렬 코퍼스 + 새로운 역 번역 데이터 0.45m BPE merge 32000의 성능

	BLEU1	BLEU2	BLEU3	BLEU4
0.2m	30.65	21.72	17.08	13.58
0.4m	30.34	22.28	17.77	14.41
0.6m	33.37	24.78	19.90	16.17
0.8m	34.80	25.86	20.81	16.95
all	36.03	26.69	21.43	17.46

[표 8] 기존 병렬 코퍼스 + 새로운 역 번역 데이터 0.98m BPE merge 32000의 성능

	BLEU1	BLEU2	BLEU3	BLEU4
0.2m	33.17	22.64	17.26	13.29
0.4m	33.53	22.62	17.54	14.43
0.6m	35.99	24.88	19.01	15.33
0.8m	34.26	25.42	20.47	16.72
all	36.60	25.83	20.77	16.95

실험 결과, 기존 병렬 코퍼스 80 만 문장 이상에 새로운 역 번역 데이터를 45 만문장 섞었을 때, BPE merge 32000 operation 을 적용시키면 성능이 가장 좋게 나왔다. 그리고 새로운 역 번역 데이터를 98 만 문장 섞은 경우에는 오히려 번역 성능이 떨어짐을 확인 할 수 있었다. 이는 새로운 데이터가 오히려 일종의 noise 처럼 작용하게 되어 성능이 떨어졌을 것으로 추정된다.

기존 병렬 코퍼스 80만 문장 이상에 새로운 역 번역 데이터를 45만문장 섞었을 때, BPE merge 32000 operation 을 적용시킨 것이 가장 성능이 좋았지만, 기존 병렬 코퍼스만 사용했을 때와 비교해보면 간혹 새로운 데이터를 추가했을 때 번역 성능이 더 좋은 경우가 있지만, 매우 미비한 수준의 성능 향상을 나타내며, 대부분은 기존의 병렬 코퍼스만 사용했을 때의 번역 성능이 더 높았다. 전반적으로 보았을 때, 단순히 역 번역을 통해 새롭게 얻어낸 병렬 코퍼스를 모두 기존의 병렬 코퍼스에 추가시키는 것은 번역 성능 향상에 도움이 되지 않음을 확인 할 수 있었다.

4.4.3 기존 병렬 코퍼스만 사용한 번역 모델과 선택적으로 성능이 높은 역 번역 데이터만 추가한 번역 모델의 성능 비교

4.4.2 에서 보였듯, 단순히 역 번역을 통해 새롭게 얻어낸 병렬 코퍼스를 모두 기존의 병렬 코퍼스에 추가시키는 것은 번역 성능 향상에 도움이 되지 않았다. 그래서, 위의 [그림 11] 처럼 WMT16 영어 단일 코퍼스를 OpenSubtitles2018 병렬 코퍼스로 학습시킨 역 번역 모델의 input 으로 넣어서 WMT16 한글 생성 코퍼스를 얻어낸 뒤, 이를 마찬가지로 OpenSubtitles2018 병렬 코퍼스로 학습시킨 Main Task 모델의 input 으로 넣어서 WMT16 영어 생성 코퍼스를 얻어낸다. 그 뒤, 얻어낸 WMT16 영어 생성 코퍼스와 기존의 WMT16 영어 단일코퍼스와의 BLEU 점수 계산을 통해, 어느정도 점수가 높은 영어 문장과 그에 상응하는 한글 문장을 새롭게 학습 데이터에 추가시키는 방법을 사용해 보았다. 아래[표 9]와 [표 10]은 얻어낸 WMT16 영어 생성 코퍼스와 기존의 WMT16 영어 단일코퍼스와의 BLEU 점수에 따라 새롭게 얻어지는 병렬 코퍼스의 수와, 그 새롭게 얻어진 병렬 코퍼스를 추가 한 뒤의 BLEU 성능을 나타낸다.

[표 9] BPE Merge 32000, BLEU 3 성능

WMT16(EN) WMT16'(KO) BLEU	> 0.01	> 10	> 20	> 30	> 40	> 50
문장수	0.91m+ 209,575	0.91m+ 203,328	0.91m+ 180,131	0.91m+ 140,635	0.91m+ 105,142	0.91m+ 79,625
BLEU 3	23.22	22.96	22.39	22.48	21.80	22.45

[표 10] BPE Merge 32000, BLEU 4 성능

WMT16(EN) WMT16'(KO) BLEU	> 0.01	> 10	> 20	> 30	> 40	> 50
문장수	0.91m+ 209,575	0.91m+ 203,328	0.91m+ 180,131	0.91m+ 140,635	0.91m+ 105,142	0.91m+ 79,625
BLEU 4	19.02	18.74	18.32	18.41	17.96	18.40

기존의 OpenSubtitles2018 병렬 코퍼스만 사용했을 때의 BPE merge 32000 의 성능은 BLEU 3 기준 21.91, BLEU 4 기준 17.93 이다.

WMT16 영어 생성 코퍼스와 기존의 WMT16 영어 단일코퍼스와 BLEU 점수를 비교하는 첫번째 구간을 0.01 로 잡은 이유는, 조금이라도 그럴듯한 번역을 해냈을 때의 기준이 BLEU 0.01 이기 때문이다. 즉, 전체 WMT16 코퍼스 98 만여 문장 중에서 조금이라도 그럴듯한 역 번역 결과가 나온 데이터는 약 21 만 문장이고, 나머지 약 78 만 문장은 원문과 전혀 연관이 없는 번역 결과가 나왔음을 의미한다.

번역 성능이 10, 20, 30, 40, 50 으로 점점 높아질 때마다, 기준을 충족시키는 새로운 생성 코퍼스의 수는 약 20 만문장, 약 18 만문장, 약 14 만문장, 약 11 만문장, 약 8 만 문장으로 점점 그 수가 줄어듦을 확인 할 수 있다.

실험 결과, 가장 성능이 높게 나온 실험은 조금이라도 그럴듯한 번역 결과가 나온 생성 코퍼스를 가장 많이 추가한 첫번째 실험이었다. 나머지 실험들도 대체로 기존 병렬코퍼스만 사용했을 때의 번역 성능보다는 높게 나타났지만, BLEU>40 구간에서 보듯이, 기존 성능보다 떨어지는 구간도 나타났다. 이를 통해, 그럴듯한 번역 결과가 나온 새로운 데이터는 대체로 많이 넣을수록 성능 향상에 도움이 되지만, 어느정도 추가되는 데이터의 양과 번역 품질 사이에 give and take 가 있는 것으로 추정이 된다.

4.4.4. 실제 번역 예시를 통한 각 모델의 번역 성능 확인

[표 11] 번역 예시 1

원문	1962 년 , 프레드릭 L . 프라이어 는 예일대학교 에서 경제학 박사학위 를 받 았 다
정답	*In 1962 , Frederic L , Pryor received his Ph , D , in Economics from Yale ,*
Baseline	<unk> <unk> , <unk> <unk> <unk> .
BPE 16000	1962 , Frederic L.L.L.P. has been in college .
BPE 32000	in 1962 , Predybert pi to the Yale University of Yale University .
BPE 32000 +BLEU>0.01	*in 1962 , Chiedrick , Pryor , Pryor had an MBA at Yale University .*

[표 12] 번역 예시 2

원문	브라트바에선 , 목적 없이 죽이 는 일 은 없 어
정답	*In Bratva , there is no death without purpose .*
Baseline	no , no , no .
BPE 16000	we don' t kill him without purpose .
BPE 32000	we don' t have to kill her .
BPE 32000 +BLEU>0.01	*Bratva , we don' t have anything to kill without a purpose .*

[표 11]과 [표 12]는 Baseline 모델과 BPE merge 16000 을 적용 시켰을 때의 모델, BPE merge 32000 을 적용 시켰을 때의 모델, 그리고 성능이 가장 좋게 나왔던 BPE merge 32000 을 적용시키고, 기존의 OpenSubtitles2018 병렬 코퍼스에 BLEU>0.01 구간에서 새롭게 얻어진 역 번역 데이터를 추가한 모델의 번역 예시를 나타낸다.

[표 11]을 보면 baseline 은 번역을 전혀 해내지 못했고, BPE merge 16000 과 BPE merge 32000 모델은 시간을 나타내는 '1962' 는 잘 번역해 내었지만,

고유명사인 '프레드릭 L. 프라이어'를 제대로 번역해 내지 못하였고, 박사학위를 나타내는 'Ph. D' 또한 제대로 번역해 내지 못하였다. 그러나 마지막 모델의 경우에는 '1962'와 '프레드릭 L. 프라이어', 를 원문과 제일 유사하게 재현해 내었고, 의미는 조금 다르지만 'MBA'라는 표현을 통해 경제학 박사의 의미를 살려내어, 가장 번역을 잘 해내었음을 확인할 수 있었다.

[표 12]를 보면 마찬가지로 baseline 모델은 번역을 전혀 해내지 못했고, BPE merge 16000 과 BPE merge 32000 모델은 원문과 의미는 유사하지만 불필요하게 대명사인 'him'이나 'her'를 추가하여 번역하였다. 역시 마지막 모델은 가장 정답과 유사한 의미를 가지도록 번역을 잘 해내었다. 번역 예시를 통해 살펴본 결과, BLEU 점수가 가장 높은 BPE merge 32000 + BLEU>0.01 모델이 실제로도 가장 번역 성능이 높음을 확인 할 수 있었다.

5. 결론 및 추후 연구

본 논문에서는 한국어-영어 기계번역에 Subword Unit 기법인 Byte Pair Encoding 을 적용시키고, Back Translation 기법의 적용을 통한 코퍼스의 확장을 통해 번역 성능을 향상시킬 수 있음을 보였다.

구체적으로는 Byte Pair Encoding 기법을 Source 언어인 한국어와 Target 언어인 영어 모두에 적용시키는 것 만으로 Baseline 모델인 한글과 영어 모두 단어를 단위로 하여 사전을 구축한 모델보다 훨씬 좋은 번역 성능을 얻어낼 수 있음을 보였고, 학습 코퍼스의 크기가 적당량 이상일 때에는 (약 80~100 만 pair) BPE merge 32000 의 성능이 더 좋고, 학습 코퍼스의 크기가 그보다 작을 때에는 BPE merge 16000 의 성능이 더 좋음을 보였다. 또한, 기존의 연구결과와는 다르게 한국어-영어 기계번역에서는 단순히 Back Translation 을 통해 얻어진 새로운 코퍼스를 모두 기존의 병렬 코퍼스에 추가하여 학습 코퍼스로 사용하는 것은 번역 성능 향상에 크게 도움이 되지 못함을 확인하였으며, 다만 Back Translation 을 통해 얻어진 새로운 생성 코퍼스 중, 번역 성능이 어느정도 되는 문장 쌍을 선별하여 기존 학습 코퍼스에 추가하면 성능 향상을 얻어낼 수 있음을 확인하였다.

향후 연구로는 코퍼스의 확장 측면에서 Back Translation 이 아닌 다른 방법을 적용시켜 보는 것이 있겠다. 예를 들자면 기존의 병렬 코퍼스를 Paraphrasing 을 통해 확장시켜 보는 것이다. 또한 이번 연구에서는 기존의 OpenSubtitles2018 데이터에 WMT16 단일 코퍼스 만을 추가하여 실험을 진행하였는데, 또다른 데이터를 더 추가했을 때에는 번역 성능이 어떻게 나올지에 대한 연구도 필요할 것이다. 또한, 이번 연구에서는 OpenSubtitles2018 데이터와는 다른 도메인을

가지는 WMT16 단일 코퍼스 만을 추가하여 실험을 진행하였는데, 같은 도메인을 가지는 데이터나 OpenSubtitles2018 데이터 자체를 다시 Back Translation 시켜서 얻어낸 새로운 병렬 코퍼스를 학습 코퍼스에 추가 시켰을 때에는 성능이 어떻게 나올지에 대한 연구도 필요할 것이다.

참 고 문 헌

[1] 황영숙, "특집 : 병렬말뭉치의 구축과 활용 ; 통계기반 기계번역 기술의 소개 및 최신 연구동향 분석", *언어사실과 관점*, 25권 0호, pp. 89-114, 2010.

[2] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to Sequence Learning with Neural Networks," In *Advances in Neural Information Processing Systems*, vol. 27, pp. 3104-3112, 2014.

[3] K. Cho, B. Merrienboer, C. Gulcehre, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation", In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pp. 1724-1734, 2014.

[4] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, "How to Construct Deep Recurrent Neural Networks", In *Proceedings of the Second International Conference on Learning Representations*, 2014.

[5] A. Graves, A. Mohamed, and G. Hinton, "Speech Recognition with Deep Recurrent Neural Networks", In *Proceedings of Acoustics, Speech and Signal Processing, 2013 IEEE International Conference*, pp. 6645-6649, 2013.

[6] S. Hochreiter, and J. Schmidhuber, "Long Short-Term Memory", *Neural Computation*, Volume 9 Issue 8, pp. 1735-1780, 1997.

[7] J. Chung, C. Gülçehre, K. cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling", *arXiv:1412.3555*, pp. 1-9, 2014.

[8] R. Sennrich, B. Haddrow, A. Birch, "Neural Machine Translation of Rare Words with Subword Units", In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, Volume 1: Long Papers, 2016.

[9] 이재환, 김보성, 허광호, 고영중, 서정연, "Subword 유닛을 이용한 영어-한국어 신경망 기계번역", *한국정보과학회 학술발표논문집*, Vol.2018 No.6, pp. 586-588, 2018.

[10] R. Sennrich, B. Haddow, and A. Birch, "Improving Neural Machine Translation Models with Monolingual Data", In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pp. 311-318, 2016.

[11] T. Luong, H. Pham, and C. D. Manning, "Effective Approaches to Attention-based Neural Machine Translation", In *proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pp. 1412-1421, 2015.

[12] D. Kingma, and J. Ba, "Adam : A method for Stochastic Optimization", In *Proceedings of the 3th International Conference on Learning Representations*, 2015.

[13] M. Denkowski, and G. Neubig, "Stronger Baselines for Trustable Results in Neural Machine Translation", In *Proceedings of the First Workshop on Neural Machine Translation, Association for Computational Linguistics*, pp. 18-27, 2017.

[14] P. Koehn, H. Hoang, A. Birch, C. C. Burch, M. Federico, N. Bertoldi, B. Cowan, W. Shen, C. Moran, R. Zens, C. Dyer, O. Bojar, A. Constantin, and E. Herbst, "Moses: Open Source Toolkit for Statistical Machine Translation", In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2007.

[15] D. Britz, A. Goldie, M. T. Luong, and Q. Le, "Massive Exploration of Neural Machine Translation Architectures", In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pp. 1442-1451, 2017.

[16] K. Papineni, S. Roukos, T. Ward, and W. J. Zhu, "BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2002.