2017 년 가을 POSTECH 컴퓨터공학과 과제연구 최종 보고서

Translation Quality Estimation Task 의 Baseline System 성능 개선

학 번: 20140165
이 름: 이소현
연구 지도교수: 이종혁
학 과: 컴퓨터공학과

연구 목적

본 연구의 목적은 translation quality estimation task 의 baseline system 인 QuEst 의 성능을 개선하여, 기계 번역 시스템의 결과물인 번역 문장의 품질을 더욱 잘 예측하도록 만드는 것이다.

연구 배경

기계 번역 기술의 발전에 따라 번역 성능에 대한 객관적인 평가 방법이 점점 더 중요해지고 있다. 기존의 평가 방법으로는 참조 문장을 이용하는 방식이나 사람이 직접 평가하는 방식이 있는데, 이들은 많은 면에서 비효율적이다. 먼저 모든 문장에 대하여 참조 문장을 이용할 수 없고, 사람이 직접 평가하는 것은 평가자에 따라 다소 주관적인 기준이 반영되어 정확하지 않을 수 있다. 또한 참조 문장을 사람이 직접 작성하거나, 많은 양의 번역 문장을 사람이 직접 평가해야 하므로 많은 시간과 노력을 필요로 한다. 따라서 참조 문장 없이 자동으로 번역 문장의 품질을 평가하고, 이를 이용해 기계 번역 모델의 성능을 평가하는 방법이 필요해졌고, 이에 따라 기계 번역을 위한 quality estimation task 가 제안되었다.

Quality estimation task 는 source 문장과 target 문장만을 이용하여 자동으로 문장 번역의 질을 평가하는 연구로, 2012 년에 시작되어 현재까지 이어지고 있다. 가장 최근에 있었던 WMT'17 Quality Estimation task 에서는 인공 신경망 기반의 다양한 접근 방법들이 제안되었고, 우수한 성능을 보인 바 있다.

QuEst 의 성능을 향상시키는 것은 번역 품질을 더욱 잘 예측하는 데에 도움을 줄 수 있어, 기계 번역 기술을 더욱 향상시킬 수 있다. 또한 번역 품질을 잘 예측하는 것은 번역에 대한 post-editing 과정에서 소요되는 시간과 노력을 경감할 수 있을 것이다.

연구 방법

이번 연구의 주된 목표는 source 문장과 target 문장에서 문장 번역의 품질을 평가하기 위해 사용할 수 있는 새로운 feature 을 찾고, 이를 QuEst 에 적용하여 성능을 향상시키는 것이다.

QuEst 는 feature extraction 을 수행하는 모듈과 machine learning 을 수행하는 모듈로 이루어져 있다. 먼저, feature extraction 모듈에서는 source 문장과 target 문장에서 학습에 필요한 feature 을 추출한다. Baseline system 에서 이용한 feature 은 총 17 개로, 다음과 같다.

- Source 문장의 총 단어 수
- Target 문장의 총 단어 수
- Source 문장의 단어들의 길이 평균
- Source 문장의 language model 확률
- Target 문장의 language model 확률
- Target 문장의 총 단어 수 / Target 문장의 unique 단어 수
- Source 문장 단어의 probability 합 / Source 문장의 총 단어 수 (probability threshold: 0.2)
- Source 문장 단어의 probability 합 / Source 문장의 총 단어 수 (probability threshold: 0.01)
- Source 문장의 unigram 중 language model 빈도수가 제 1 사분위수 이하인 unigram 의 비율 (빈도수가 낮은 unigram)
- Source 문장의 unigram 중 language model 빈도수가 제 3 사분위수 이상인 unigram 의 비율 (빈도수가 높은 unigram)
- Source 문장의 bigram 중 language model 빈도수가 제 1 사분위수 이하인 bigram 의 비율
- Source 문장의 bigram 중 language model 빈도수가 제 3 사분위수 이상인 bigram 의 비율
- Source 문장의 trigram 중 language model 빈도수가 제 1 사분위수 이하인 trigram 의 비율
- Source 문장의 trigram 중 language model 빈도수가 제 3 사분위수 이상인 trigram 의 비율

- SMT training corpus 에 존재하는 source 문장의 unique unigram 개수 / source 문장의 unique unigram 개수
- Source 문장의 구두점 개수
- Target 문장의 구두점 개수

Machine learning 모듈에서는 Support Vector Machine (SVM)을 이용하여 추출된 feature 들과 그에 대응하는 번역 평가 점수를 학습하여 번역 문장의 품질을 예측한다. 예측에 대한 성능 평가 방법으로는 Mean Absolute Error (MAE)와 Root Mean Squared Error (RMSE)를 이용한다.

Dataset 은 WMT'12 Quality Estimation (QE) dataset 의 문장 단위의 영어-스페인어 번역문 및 번역 평가 점수를 이용하였다. Source 문장의 언어는 영어이고 target 문장의 언어는 스페인어이며 번역 평가 점수는 1.0 점에서 5.0 점 사이의 소수점 첫째자리까지의 실수로 이루어져있다. Training data 는 영어 source 문장, 스페인어 target 문장, 번역 평가 점수 각 1832 개씩으로 구성되어 있고, testing data 는 source, target 문장 각각 422 개씩으로 이루어져있다. Testing data 에 대한 번역 평가 점수 또한 제공되어 있다.

QuEst 에는 baseline system 이 사용한 17 가지의 feature 외에도 60 개의 feature 이 주어져 있다. 먼저, 이들을 모두 feature 목록에 추가하여 77 개의 feature 로 번역 문장 품질을 예측해보았다. 또 overfitting 과 training 시간을 줄이고 정확도를 높이기 위하여 77 개의 feature 중에 몇가지 feature 를 선택하여 품질을 예측해보기도 하였다. Feature selection 의 method 로 Random Lasso 를 이용하였는데, Random lasso 는 밀접하게 연관되어 있는 feature 사이에서의 feature selection 에 이점을 가진다고 알려져 있다 [Wang et al. (2011)].

새롭게 고안한 feature 들은 QuEst baseline system 의 단점을 보완하기 위하여 문장의 의미적인 요소를 고려하여 구상하였다. QuEst 에 존재하는 기존의 feature 들은 source 문장과 target 문장에 대한 각각의 단어 개수나 N-gram 비율, probability 등 문장에 대한 문법적인 요소로만 이루어져있다. 번역 문장 품질을 평가하는 데에 있어서 문장의 의미는 매우 중요한 영향을 끼치는데, source 문장과 target 문장의 의미에 대한 평가는 존재하지 않는 것이다. 따라서 문장의 의미에 대한 feature 를 추출하기로 결정하였고, 기존 QuEst 의 feature 추출 방법을 이용하는 것과 neural network 를 이용하는 두가지 방법으로 feature 들을 추출하였다.

첫번째로는 문장에서의 복잡한 단어의 비율을 이용하여 3 가지의 feature 을 새롭게 고안하였다. 복잡한 단어일수록 더 많은 의미를 담고 있기 때문에 이들이 잘 유지되는 것만으로도 대체적인 의미가 통할 것이라고 생각했다. 이 의미를 직접적으로 비교하지는

않고 복잡한 단어가 들어있는 비율을 feature 로 추출하였다. 이를 위한 resource 로 먼저 Wiktionary 의 1000 basic English words 목록을 이용하여 영어에 대한 간단한 단어 목록을 먼저 구성하였다. 이후 번역기를 이용하여 이에 대응하는 스페인어 간단한 단어 목록을 구성하여 두 목록의 단어 의미가 일치하도록 하였다. Source 문장과 target 문장에서 간단한 단어 목록에 없는 단어의 개수 세어 전체 단어에 대한 비율을 계산하였고, 이를 통해 만든 feature 은 다음과 같다.

- Source 문장의 복잡한 단어 개수 / source 문장의 총 단어 개수 (Feature7001)
- Target 문장의 복잡한 단어 개수 / target 문장의 총 단어 개수 (Feature7002)
- Feature 7001 의 값 / Feature7002 의 값 (Feature7003)

두번째로는 문장의 직접적인 의미를 feature 로 나타내기 위하여, 문장을 하나의 vector 로 나타내어 그 vector 의 element 를 각각의 feature 들로 추가하였다. 문장을 하나의 vector 로 나타내기 위해서 word2vec 와 autoencoder 를 이용하였다.

Word2vec 는 단어를 단어의 의미에 맞게 좌표 상의 vector 로 바꾸어주는 함수이다. 영어와 스페인어 각각에 대한 word2vec 가 필요하고, word2vec 를 구성하기 위한 기계 학습을 위해서 영어와 스페인어의 문장 dataset 이 필요하다. 문장 dataset 으로 Europarl Parallel Corpus version 7 의 영어-스페인어 parallel corpus 를 사용하였고, dataset 에 10 번 이상 등장한 단어들에 대하여 word2vec 를 구성하였다. Word2vec 의 vector dimension 은 30 으로 설정하였으며, dataset 에서 최종적으로 학습한 영어 word2vec 의 총 단어 수는 30802 개, 스페인어 word2vec 의 총 단어 수는 48258 개이다. 이 단어들은 WMT'12 QE training data 총 단어 집합의 약 80.6 %의 단어를 포함하고 있다.

만든 word2vec 를 이용하여 문장의 단어들을 모두 vector 로 표현하고 이들을 단어 순서대로 일렬로 나열하면, (문장 단어 수 * word vector dimension)를 dimension 으로 하는 vector 를 만들 수 있다. 이 때 이 sentence vector 에 대한 각각의 차원들의 값을 feature 로 추가하면 학습하기에는 너무나도 많은 feature 가 생성된다. 따라서 sentence vector 의 dimension 을 학습하기에 효율적인 값만큼 낮추어야 하고, 이를 위해 autoencoder 를 사용하였다.

Autoencoder 는 input 으로 임의의 값을 주면 hidden state 로 input 을 압축하였다가 output 으로 input 과 같은 값을 내보내는 neural network model 이다. 이 model 의 input 을 각 문장의 word vector 를 합친 vector 로 하여 학습시키고, 학습된 hidden state 의 값을 문장을 나타내는

vector 로 사용하는 것이다. WMT'12 QE training data 를 이 model 의 training data input 으로 사용하여 영어와 스페인어 각각의 autoencoder model 을 만들었다. Input 문장의 길이가 제각기 다르기 때문에 input 크기를 동일하게 맞추기 위하여, 각 문장의 길이를 가장 긴 문장의 길이와 같도록 0 으로 padding 을 해준 후, input 으로 사용하였다. 각 언어 model 은 training data 에 대하여 20 epoch 씩 training 하였다. Hidden state 의 dimension 을 4, 8, 16 으로 바꾸어가며 영어, 스페인어를 합쳐 총 8, 16, 32 개의 feature 를 추출하였다. Model 의 loss function 은 $(input - output)^2$으로 하였으며, model 에 대한 loss 값은 아래 표 1. 과 같다.

Autoencoder model	Loss	Autoencoder model	Loss
영어 dim 4	0.0074	스페인어 dim 4	0.0072
영어 dim 8	0.0074	스페인어 dim 8	0.0073
영어 dim16	0.0075	스페인어 dim 16	0.0073

표 1. 각 Autoencoder model 의 loss 값

연구 결과 및 평가

Baseline system 에 대한 논문 상에서의 baseline system 의 성능은 MAE 가 0.6802, RMSE 가 0.8192 이다. 하지만 실제 환경 상에서의 baseline system 성능은 MAE 가 0.7070, RMSE 가 0.8564 로 논문의 성능과 오차가 존재한다. 이는 baseline system 에 존재하는 feature 를 구하는 데에 사용되는 language model resource 가 현재 설치된 baseline system 의 language model resource 와 다른 것에서 비롯되었다고 생각하였다. 따라서 현재의 system 에 맞게 현재 환경의 성능을 baseline (BL)으로 선택하였다. Baseline system 과 새롭게 만든 system 들에 대한 성능 비교 결과는 아래 표 2. 와 같다.

QuEst 에 존재하는 60 개의 feature 를 baseline 에 추가하여 총 77 개의 feature 로 학습한 결과, baseline 의 성능보다 MAE 와 RMSE 가 증가하였다. 문장을 나타내는 feature 이더라도 그 성능이 기존의 feature 보다 좋지 않으면, 많이 추가하더라도 성능을 떨어뜨리는 요인이 된다는 것을 알 수 있었다. 이 중에서 좋은 성능의 feature 를 선택하였을 때에는 좋은 결과를 가져올지 알아보기 위하여, 77 개의 feature 에서 random lasso (RL)를 사용하여 몇가지 feature 를 선택해보았다. 이 때, random lasso 의 selection score threshold 를 조정해가며 여러가지 feature 집합을 만들어 성능을 측정해보았다. Feature 를 줄이면 random lasso 의 기준에서 좋은 feature 들이 선택되므로 더 성능이 오를 것이라고 추측하였는데, 예상과 달리

feature 수가 줄수록 성능이 더욱 낮아졌다. 함께 있을 때 좋은 성능을 내는 feature 짝이 있을 텐데, 떨어져 있어서 성능이 좋지 않아 선택되지 않은 feature 가 있을 것이라는 결론을 내렸다.

Baseline 에 Feature7001, Feature7002, Feature7003 를 추가한 후의 성능 또한 낮아졌다. 먼저 간단한 단어의 목록의 출처가 Wiktionary 의 1000 basic English words 였고, 이 목록의 간단한 단어라는 기준이 애매모호하여 성능이 좋은 feature 를 찾아내지 못하였다. 또한 단순히 복잡한 단어의 비율을 계산하는 것으로는 문장의 의미에 대한 적합한 feature 를 추출할 수 없다는 것을 알 수 있다.

마지막으로 sentence vector (SV)를 이용하여 feature 를 추출하였을 때, baseline system 보다 MAE 면에서 성능이 증가한 system 을 만들 수 있었다. 성능 평가 결과를 통해 vector 의 dimension 이 8 일 때 성능이 가장 좋다는 것을 알 수 있다. Vector 의 dimension 을 4 로 하였을 때에는 baseline 보다 성능이 항상 낮은데, dimension 이 너무 낮아 문장의 특징을 잘 압축하지 못하였을 것이다. Dimension 이 8 일 때는 random lasso 를 통하여 29 개의 feature 가 선택되었을 때 MAE 와 RMSE 두 쪽 모두 가장 좋은 성능을 냈다. 적절한 개수의 중요하지 않은 feature 들이 버려지고 남은 feature 들로 비교적 정확한 평가 점수를 예측할 수 있었다. Vector 의 dimension 이 16 이면 문장의 특징이 더 자세히 저장되어 있어 더 좋은 성능을 낼 것 같았지만, 예상외로 그렇지 않았다. Baseline 보다 성능이 좋아지긴 하였지만, feature 의 개수가 증가하여 번역 평가 점수와 feature 사이의 연관성을 찾는 SVM 학습에 부정적인 영향을 끼쳤을 것이라고 생각하였다.

구상한 모든 system 의 성능 평가 결과, baseline 에 dimension 8 의 sentence vector 를 feature 로 추가하고 random lasso 로 29 개를 선택하였을 때 가장 좋은 성능을 내었다.

구현 코드는 다음 URL 에서 확인할 수 있다. https://github.com/lshhhhh/questplusplus

System (number of features)	MAE	RMSE
BL (17)	**0.7070**	**0.8564**
BL + 60 features (77)	0.7176	0.8756
BL + 60 features → RL (63)	0.7176	0.8756
BL + 60 features → RL (54)	0.7163	0.8751
BL + 60 features → RL (54)	0.7225	0.8783
BL + 60 features → RL (25)	0.7251	0.8802
Feature 7001, 7002 (2)	0.8123	0.9841
Feature 7001, 7002, 7003 (3)	0.8059	0.9699
BL + 7001, 7002 (19)	0.7088	0.8588
BL + 7001, 7002, 7003(20)	0.7103	0.8598

SV4 (8)	0.8064	0.9756
BL + SV4 (25)	0.7105	0.8684
BL + SV4 → RL (21)	0.7165	0.8729
SV8 (16)	0.7876	0.9607
BL + SV8 (33)	**0.7052**	0.8605
BL + SV8 → RL (32)	0.7085	0.8663
BL + SV8 → RL (29)	**0.7001**	0.8565
BL + SV8 → RL (25)	**0.7026**	0.8593
SV16 (32)	0.7955	0.9564
BL + SV16 (49)	**0.7062**	0.8583
BL + SV16 → RL (43)	**0.7059**	0.8586
BL + SV16 → RL (33)	**0.7068**	0.8587
BL + SV16 → RL (23)	0.7169	0.8661

표 2. 각 System 의 성능 평가 결과

토론 및 전망

QuEst 에는 존재하지 않는 의미적인 요소를 가진 feature 를 추가함으로써 baseline 의 성능을 증가시킬 수 있었다. 새롭게 구상한 feature 들의 더 나은 성능을 위하여 다음 항목들을 보완하면 좋을 것이다.

Word2vec 를 구성할 때, 더 많은 문장 data 를 이용하여 training 하면 더욱 질 좋은 word2vec 를 구성할 수 있다. 또한 word vector 의 dimension 을 조정해보며 feature 추출을 위한 최적의 dimension 을 찾으면 더 좋은 성능의 feature 를 만들 수 있을 것이다.

Autoencoder 를 구성할 때에도 더 많은 문장 data 를 이용하여 training 하면 autoencoder 의 성능을 올릴 수 있다. 지금의 model 은 한 층의 hidden state 만이 존재하는데, hidden state 의 층과 개수를 변형하며 최적의 model 을 찾을 수도 있을 것이다. 또한 autoencoder 의 input 으로 문장을 넣을 때, 문장 길이를 맞추기 위하여 padding 을 이용하였는데 이것이 문장의 의미를 뽑는데 악영향을 주었을 것이다. 이번 연구에서는 시간 제약에 의해 간단한 구현과 빠른 결과 확인을 위하여 sentence vector 를 구성하는 데에 autoencoder 를 사용하였다. Autoencoder 보다 비교적 학습 속도가 느린 RNN encoder 로도 sentence vector 를 구성할 수 있다. RNN 기법에서는 input 의 길이가 일정하지 않아도 되는 dynamic RNN 을 이용할 수 있으므로 padding 이 필요 없는 encoder 를 만들 수 있을 것이다.

Machine learning module 에서 사용한 SVM 기법은 인공 신경망 기법에 비해 비교적 성능이 안 좋을 수 있지만, 빠르게 학습하고 사용할 수 있다는 장점이 있다. 또한 QuEst 에서 사용되는 feature 은 인공 신경망 기법에도 적용할 수 있으므로, 이번 연구의 결과는 인공 신경망 quality estimation 의 성능을 올리는 데에도 도움이 될 것이다.

참고 문헌

Shah, K., Avramidis, E., Biçici, E., & Specia, L. (2013). QuEst – Design, Implementation and Extensions of a Framework for Machine Translation Quality Estimation. *The Prague Bulletin of Mathematical Linguistics, 100*. doi:10.2478/pralin-2013-0008

Wang, S., Nan, B., Rosset, S., & Zhu, J. (2011). Random lasso. *The Annals of Applied Statistics, 5*(1), 468-485. doi:10.1214/10-aoas377

Tsvetkov, Y., Faruqui, M., & Dyer, C. (2016). Correlation-based Intrinsic Evaluation of Word Vector Representations. *Proceedings of the 1st Workshop on Evaluating Vector-Space Representations for NLP*. doi:10.18653/v1/w16-2520

Liou, C., Cheng, W., Liou, J., & Liou, D. (2014). Autoencoder for words. *Neurocomputing, 139*, 84-96. doi:10.1016/j.neucom.2013.09.055

EMNLP 2017 second conference on machine translation. http://www.statmt.org/wmt17/quality-estimation-task.html

QuEst - an open source tool for translation quality estimation. http://staffwww.dcs.shef.ac.uk/people/L.Specia/projects/quest.html

Europarl Parallel Corpus. http://www.statmt.org/europarl/