Sequence to Sequence Learning with Neural Networks

Ilya Sutskever
Google
ilyasu@google.com

Oriol Vinyals
Google
vinyals@google.com

Quoc V. Le
Google
qvl@google.com

배성은

(0) Abstract

Abstract

Deep Neural Networks (DNNs) are powerful models that have achieved excellent performance on difficult learning tasks. Although DNNs work well whenever large labeled training sets are available, they cannot be used to map sequences to sequences. In this paper, we present a general end-to-end approach to sequence learning that makes minimal assumptions on the sequence structure. Our method uses a multilayered Long Short-Term Memory (LSTM) to map the input sequence to a vector of a fixed dimensionality, and then another deep LSTM to decode the target sequence from the vector. Our main result is that on an English to French translation task from the WMT-14 dataset, the translations produced by the LSTM achieve a BLEU score of 34.8 on the entire test set, where the LSTM's BLEU score was penalized on out-of-vocabulary words. Additionally, the LSTM did not have difficulty on long sentences. For comparison, a phrase-based SMT system achieves a BLEU score of 33.3 on the same dataset. When we used the LSTM to rerank the 1000 hypotheses produced by the aforementioned SMT system, its BLEU score increases to 36.5, which is close to the previous state of the art. The LSTM also learned sensible phrase and sentence representations that are sensitive to word order and are relatively invariant to the active and the passive voice. Finally, we found that reversing the order of the words in all source sentences (but not target sentences) improved the LSTM's performance markedly, because doing so introduced many short term dependencies between the source and the target sentence which made the optimization problem easier.

DNN은 매우 뛰어난 성능

그러나 시퀀스를 시퀀스에 매핑하는데 사용할 수 없다.

이 논문에선,
시퀀스학습에 대한 일반적인 end-to-end 접근방식을 제시

다층 LSTM을 사용해 입력 시퀀스를 고정된 차원의 벡터에 매핑한 다음 또 다른 심층 LSTM을 사용해 벡터에서 대상 시퀀스를 디코딩한다.
2개의 LSTM을 사용해 시퀀스 문제를 해결하고자 했음.
높은 BLEU 점수 달성함

모든 원문 문장(대상문장이 아님)에서 단어의 순서를 뒤집으면 LSTM의 성능이 현저하게 향상되는 것을 발견.
순서를 뒤집는 것은 대상문장 사이에 많은 단기 의존성이 도입되어 최적화 문제가 더 쉬워지기 때문이다.

(0) Abstract

BLEU (Bilingual Evaluation Understudy) 점수란?

기계번역 결과와 사람이 직접 번역한 결과가 얼마나 유사한지 비교하여 번역에 대한 성능을 측정하는 방법
측정 기준은 n_gram에 기반

완벽한 방법은 아니지만, 몇가지 이점이 있음
- 언어에 구애받지 않음
- 계산 속도가 빠름
- 높을 수록 성능이 더 좋음 -> 직관적

1) 단어 개수 카운트로 측정 (Unigram Precision)

$$\text{Unigram Precision} = \frac{\text{Ref들 중에서 존재하는 Ca의 단어의 수}}{\text{Ca의 총 단어 수}} = \frac{\text{the number of Ca words(unigrams) which occur in any Ref}}{\text{the total number of words in the Ca}}$$

https://wikidocs.net/31695

(1) Introduction

1 Introduction

Deep Neural Networks (DNNs) are extremely powerful machine learning models that achieve excellent performance on difficult problems such as speech recognition [13, 7] and visual object recognition [19, 6, 21, 20]. DNNs are powerful because they can perform arbitrary parallel computation for a modest number of steps. A surprising example of the power of DNNs is their ability to sort N N-bit numbers using only 2 hidden layers of quadratic size [27]. So, while neural networks are related to conventional statistical models, they learn an intricate computation. Furthermore, large DNNs can be trained with supervised backpropagation whenever the labeled training set has enough information to specify the network's parameters. Thus, if there exists a parameter setting of a large DNN that achieves good results (for example, because humans can solve the task very rapidly), supervised backpropagation will find these parameters and solve the problem.

Despite their flexibility and power, DNNs can only be applied to problems whose inputs and targets can be sensibly encoded with vectors of fixed dimensionality. It is a significant limitation, since many important problems are best expressed with sequences whose lengths are not known a-priori. For example, speech recognition and machine translation are sequential problems. Likewise, question answering can also be seen as mapping a sequence of words representing the question to a sequence of words representing the answer. It is therefore clear that a domain-independent method that learns to map sequences to sequences would be useful.

DNN은 매우 뛰어난 성능 (병렬계산과 정렬능력 덕분)

그러나, DNN은 고정된 차원의 벡터로 입력과 대상을 적절하게 인코딩할 수 있는 문제에만 적용 가능
예를 들어, 음성인식과 기계 번역 같은 순차적인 문제에 대해서 어려움.
길이가 알려지지 않은 시퀀스는 많은 문제가 생겨서 제한사항임

(1) Introduction

Sequences pose a challenge for DNNs because they require that the dimensionality of the inputs and outputs is known and fixed. In this paper, we show that a straightforward application of the Long Short-Term Memory (LSTM) architecture [16] can solve general sequence to sequence problems. The idea is to use one LSTM to read the input sequence, one timestep at a time, to obtain large fixed-dimensional vector representation, and then to use another LSTM to extract the output sequence from that vector (fig. 1). The second LSTM is essentially a recurrent neural network language model [28, 23, 30] except that it is conditioned on the input sequence. The LSTM's ability to successfully learn on data with long range temporal dependencies makes it a natural choice for this application due to the considerable time lag between the inputs and their corresponding outputs (fig. 1).

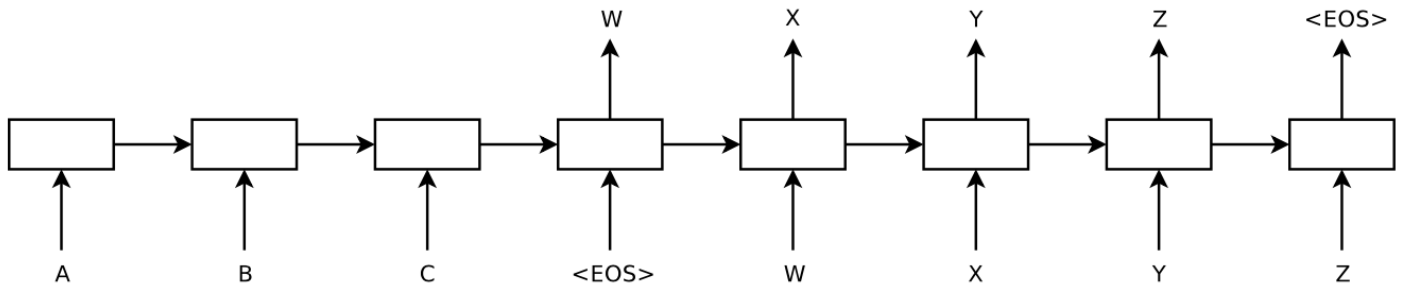


Figure 1: Our model reads an input sentence "ABC" and produces "WXYZ" as the output sentence. The model stops making predictions after outputting the end-of-sentence token. Note that the LSTM reads the input sentence in reverse, because doing so introduces many short term dependencies in the data that make the optimization problem much easier.

DNN 은 고정된 크기의 차원인 벡터인 input과 target에 대해 encode 할 수 있어서 문제가 된다.

LSTM은 시퀀스-시퀀스 문제를 해결할 수 있음
하나의 LSTM을 사용하여 한 번에 한 단계씩 입력 시퀀스를 읽어 큰 고정차원 벡터표현을 얻은 다음,
다른 LSTM을 사용하여 해당 벡터에서 출력 시퀀스를 추출하는 것.
LSTM은 입력 문장을 반대로 읽음 → 단기의존성이 도입되어서 최적화 문제를 훨씬 쉽게 만드는 데이터가 됨.

(1) Introduction

The main result of this work is the following. On the WMT'14 English to French translation task, we obtained a BLEU score of **34.81** by directly extracting translations from an ensemble of 5 deep LSTMs (with 380M parameters each) using a simple left-to-right beam-search decoder. This is by far the best result achieved by direct translation with large neural networks. For comparison, the BLEU score of a SMT baseline on this dataset is 33.30 [29]. The 34.81 BLEU score was achieved by an LSTM with a vocabulary of 80k words, so the score was penalized whenever the reference translation contained a word not covered by these 80k. This result shows that a relatively unoptimized neural network architecture which has much room for improvement outperforms a mature phrase-based SMT system.

34.81 점수로

개선의 여지가 많은 것을 보여줌.

상대적으로 최적화 되어있지 않은 신경망 아키텍처가 성숙한 구문 기반 SMT 시스템보다 성능이 우수함을 보여줌.

Finally, we used the LSTM to rescore the publicly available 1000-best lists of the SMT baseline on the same task [29]. By doing so, we obtained a BLEU score of 36.5, which improves the baseline by 3.2 BLEU points and is close to the previous state-of-the-art (which is 37.0 [9]).

Surprisingly, the LSTM did not suffer on very long sentences, despite the recent experience of other researchers with related architectures [26]. We were able to do well on long sentences because we reversed the order of words in the source sentence but not the target sentences in the training and test set. By doing so, we introduced many short term dependencies that made the optimization problem much simpler (see sec. 2 and 3.3). As a result, SGD could learn LSTMs that had no trouble with long sentences. The simple trick of reversing the words in the source sentence is one of the key technical contributions of this work.

LSTM은 매우 긴 문장에서도 문제가 없었음.

학습 및 테스트 세트에서 대상 문장이 아닌 원문 문장의 단어 순서를 뒤집었기때문에 긴 문장에서 잘 할 수 있었다.

최적화 문제를 더 간단하게 만드는 단기 종속성을 도입했음.
SGD는 긴 문장에 문제가 없는 LSTM을 학습할 수 있었다.

A useful property of the LSTM is that it learns to map an input sentence of variable length into a fixed-dimensional vector representation. Given that translations tend to be paraphrases of the source sentences, the translation objective encourages the LSTM to find sentence representations that capture their meaning, as sentences with similar meanings are close to each other while different sentences meanings will be far. A qualitative evaluation supports this claim, showing that our model is aware of word order and is fairly invariant to the active and passive voice.

LSTM의 유용한 속성은 가변길이의 입력 문장을 고정 차원 벡터 표현으로 매핑하는 방법을 학습한다는 것.

(2) model

2 The model

The Recurrent Neural Network (RNN) [31, 28] is a natural generalization of feedforward neural networks to sequences. Given a sequence of inputs $(x_1, \ldots, x_T)$, a standard RNN computes a sequence of outputs $(y_1, \ldots, y_T)$ by iterating the following equation:

$$\begin{aligned} h_t &= \text{sigm}\left(W^{\text{hx}} x_t + W^{\text{hh}} h_{t-1}\right) \\ y_t &= W^{\text{yh}} h_t \end{aligned}$$

The RNN can easily map sequences to sequences whenever the alignment between the inputs the outputs is known ahead of time. However, it is not clear how to apply an RNN to problems whose input and the output sequences have different lengths with complicated and non-monotonic relationships.

A simple strategy for general sequence learning is to map the input sequence to a fixed-sized vector using one RNN, and then to map the vector to the target sequence with another RNN (this approach has also been taken by Cho et al. [5]). While it could work in principle since the RNN is provided with all the relevant information, it would be difficult to train the RNNs due to the resulting long term dependencies [14, 4] (figure 1) [16, 15]. However, the Long Short-Term Memory (LSTM) [16] is known to learn problems with long range temporal dependencies, so an LSTM may succeed in this setting.

RNN 특징 및 한계
- RNN은 입력과 출력 간의 정렬이 미리 알려질 때마다 시퀀스를 시퀀스에 쉽게 매핑할 수 있음.
- 입출력 시퀀스가 서로 다른 길이를 가지고 있는 문제에서 적용 방법이 명확하지 않음.
- 일반적인 시퀀스 학습을 위한 방법은 하나의 RNN 사용하여 입력 시퀀스를 고정된 크기의 벡터에 매핑한 다음 벡터를 다른 RNN을 사용하여 대상 시퀀스에 매핑하는 것이다.
➔결과적인 장기 종속성으로 인해 RNN을 훈련시키는 것은 어려울 것.

그러나 LSTM은 긴 범위의 시간적 종속성이 있는 문제를 학습하는 것으로 알려져 있으므로
이 설정에서 성공할 수 있었음.

(2) model

The goal of the LSTM is to estimate the conditional probability $p(y_1, \ldots, y_{T'}|x_1, \ldots, x_T)$ where $(x_1, \ldots, x_T)$ is an input sequence and $y_1, \ldots, y_{T'}$ is its corresponding output sequence whose length T' may differ from T. The LSTM computes this conditional probability by first obtaining the fixed-dimensional representation v of the input sequence $(x_1, \ldots, x_T)$ given by the last hidden state of the LSTM, and then computing the probability of $y_1, \ldots, y_{T'}$ with a standard LSTM-LM formulation whose initial hidden state is set to the representation v of $x_1, \ldots, x_T$:

$$p(y_1, \ldots, y_{T'}|x_1, \ldots, x_T) = \prod_{t=1}^{T'} p(y_t|v, y_1, \ldots, y_{t-1}) \quad (1)$$

In this equation, each $p(y_t|v, y_1, \ldots, y_{t-1})$ distribution is represented with a softmax over all the words in the vocabulary. We use the LSTM formulation from Graves [10]. Note that we require that each sentence ends with a special end-of-sentence symbol "<EOS>", which enables the model to define a distribution over sequences of all possible lengths. The overall scheme is outlined in figure 1, where the shown LSTM computes the representation of "A", "B", "C", "<EOS>" and then uses this representation to compute the probability of "W", "X", "Y", "Z", "<EOS>".

Our actual models differ from the above description in three important ways. First, we used two different LSTMs: one for the input sequence and another for the output sequence, because doing so increases the number model parameters at negligible computational cost and makes it natural to train the LSTM on multiple language pairs simultaneously [18]. Second, we found that deep LSTMs significantly outperformed shallow LSTMs, so we chose an LSTM with four layers. Third, we found it extremely valuable to reverse the order of the words of the input sentence. So for example, instead of mapping the sentence a, b, c to the sentence α, β, γ, the LSTM is asked to map c, b, a to α, β, γ, where α, β, γ is the translation of a, b, c. This way, a is in close proximity to α, b is fairly close to β, and so on, a fact that makes it easy for SGD to "establish communication" between the input and the output. We found this simple data transformation to greatly boost the performance of the LSTM.

LSTM의 목표는 조건부 확률을 추정하는 것

식의 p는 모든 단어에 대한 softmax로 표현

실제 모델은 세가지가 다르다.

첫째, 두개의 서로 다른 LSTM사용
하나는 입력 시퀀스용, 하나는 출력 시퀀스용
→ 사소한 cost정도의 model parameter 수 증가
→ 동시에 다양한 언어 문장들에 대한 LSTM학습 가능

둘째, deep LSTM이 얕은 LSTM보다 성능이 뛰어난 걸 확인해 4개의 layer가 있는 LSTM을 선택함

셋째, 입력 문장의 단어 순서를 반대로 하는 것이 매우 유용한 것을 깨달음.

(3) Experiments

3 Experiments

We applied our method to the WMT'14 English to French MT task in two ways. We used it to directly translate the input sentence without using a reference SMT system and we it to rescore the n-best lists of an SMT baseline. We report the accuracy of these translation methods, present sample translations, and visualize the resulting sentence representation.

영어-프랑스어 MT작업에 두가지 방식 적용

참조 SMT 시스템을 사용하지 않고 입력문장을 직접 번역하는데 사용했으며, SMT기준선의 n-베스트 목록을 다시 채점.
이러한 번역의 정확성을 보고하고, 샘플 번역을 제시하고, 결과 문장 표현을 시각화함.

3.1 Dataset details

We used the WMT'14 English to French dataset. We trained our models on a subset of 12M sentences consisting of 348M French words and 304M English words, which is a clean "selected" subset from [29]. We chose this translation task and this specific training set subset because of the public availability of a tokenized training and test set together with 1000-best lists from the baseline SMT [29].

3.1. 데이터셋 세부정보

WMT'14 영어 to 프랑스어 데이터셋 사용

As typical neural language models rely on a vector representation for each word, we used a fixed vocabulary for both languages. We used 160,000 of the most frequent words for the source language and 80,000 of the most frequent words for the target language. Every out-of-vocabulary word was replaced with a special "UNK" token.

두 언어 모두에 대해 고정된 어휘를 사용(일반적인 신경언어모델을 각 단어 벡터 표현에 의존함으로)

모든 어휘 외 단어는 "UNK"토큰으로 대체

(3) Experiments

3.2 Decoding and Rescoring

The core of our experiments involved training a large deep LSTM on many sentence pairs. We trained it by maximizing the log probability of a correct translation T given the source sentence S, so the training objective is

$$1/|\mathcal{S}| \sum_{(T,S)\in\mathcal{S}} \log p(T|S)$$

where $\mathcal{S}$ is the training set. Once training is complete, we produce translations by finding the most likely translation according to the LSTM:

$$\hat{T} = \arg\max_{T} p(T|S) \quad (2)$$

Method	test BLEU score (ntst14)
Bahdanau et al. [2]	28.45
Baseline System [29]	33.30
Single forward LSTM, beam size 12	26.17
Single reversed LSTM, beam size 12	30.59
Ensemble of 5 reversed LSTMs, beam size 1	33.00
Ensemble of 2 reversed LSTMs, beam size 12	33.27
Ensemble of 5 reversed LSTMs, beam size 2	34.50
Ensemble of 5 reversed LSTMs, beam size 12	**34.81**

Table 1: The performance of the LSTM on WMT'14 English to French test set (ntst14). Note that an ensemble of 5 LSTMs with a beam of size 2 is cheaper than of a single LSTM with a beam of size 12.

3.2. 디코딩 및 재채점

실험 핵심은 많은 문장 쌍에 대한 LSTM훈련 올바른 번역의 로그확률을 최대화하여 학습함.

T : a correct Translation
S : source sentence

학습이 완료되면 가장 가능성이 높은 번역을 찾아 번역을 생성한다.

Left-to-right beam search decoder 사용
Greedy Decoding 방식 : 단순하게 해당 시점에서 가장 높은 확률이 높은 후보를 선택하는 것. 이런 경우 한번이라도 틀린 예측이 나오면, 디코딩 방식에선 치명적 문제가 됨.

Beam Search는 위 방식 보안해서 나온 방식
해당 시점에서 유망한 빔의 개수만큼 골라서 진행하는 방식

(3) Experiments

3.3 Reversing the Source Sentences

While the LSTM is capable of solving problems with long term dependencies, we discovered th the LSTM learns much better when the source sentences are reversed (the target sentences are n reversed). By doing so, the LSTM's test perplexity dropped from 5.8 to 4.7, and the test BLE scores of its decoded translations increased from 25.9 to 30.6.

While we do not have a complete explanation to this phenomenon, we believe that it is caused b the introduction of many short term dependencies to the dataset. Normally, when we concatenate source sentence with a target sentence, each word in the source sentence is far from its correspondir word in the target sentence. As a result, the problem has a large "minimal time lag" [17]. B reversing the words in the source sentence, the average distance between corresponding words the source and target language is unchanged. However, the first few words in the source languag are now very close to the first few words in the target language, so the problem's minimal time lag greatly reduced. Thus, backpropagation has an easier time "establishing communication" betwee the source sentence and the target sentence, which in turn results in substantially improved overa performance.

Initially, we believed that reversing the input sentences would only lead to more confident predi tions in the early parts of the target sentence and to less confident predictions in the later parts. Hov ever, LSTMs trained on reversed source sentences did much better on long sentences than LSTM trained on the raw source sentences (see sec. 3.7), which suggests that reversing the input sentences results in LSTMs with better memory utilization.

3.3 원문 문장 역순

입력 시퀀스의 순서를 바꿈으로써, 혼잡도는 낮아지고
BLEU 는 증가

그 이유는
Dataset의 많은 단기의존성이 도입되었기 때문
설명은 없지만,

원천 문장을 뒤집어도 source word와 target word 사이 거리의 평균은 동일하게 유지되지만,

원천 문장과 대상 문장 간의 minimal time lag가 줄어들어서 역전파에서도 더 학습이 원활해짐.

minimal time lag : 역전파 학습과정에서 기울기 소실에 의해서 time lag가 길어지는 상황

(3) Experiments

3.4 Training details

We found that the LSTM models are fairly easy to train. We used deep LSTMs with 4 layers, with 1000 cells at each layer and 1000 dimensional word embeddings, with an input vocabulary of 160,000 and an output vocabulary of 80,000. We found deep LSTMs to significantly outperform shallow LSTMs, where each additional layer reduced perplexity by nearly 10%, possibly due to their much larger hidden state. We used a naive softmax over 80,000 words at each output. The resulting LSTM has 380M parameters of which 64M are pure recurrent connections (32M for the "encoder" LSTM and 32M for the "decoder" LSTM). The complete training details are given below:

- We initialized all of the LSTM's parameters with the uniform distribution between -0.08 and 0.08
- We used stochastic gradient descent without momentum, with a fixed learning rate of 0.7. After 5 epochs, we begun halving the learning rate every half epoch. We trained our models for a total of 7.5 epochs.
- We used batches of 128 sequences for the gradient and divided it the size of the batch (namely, 128).
- Although LSTMs tend to not suffer from the vanishing gradient problem, they can have exploding gradients. Thus we enforced a hard constraint on the norm of the gradient [10, 25] by scaling it when its norm exceeded a threshold. For each training batch, we compute $s = \|g\|_2$, where g is the gradient divided by 128. If $s > 5$, we set $g = \frac{5g}{s}$.
- Different sentences have different lengths. Most sentences are short (e.g., length 20-30) but some sentences are long (e.g., length > 100), so a minibatch of 128 randomly chosen training sentences will have many short sentences and few long sentences, and as a result, much of the computation in the minibatch is wasted. To address this problem, we made sure that all sentences within a minibatch were roughly of the same length, which a 2x speedup.

3.4. 학습 상세

LSTM 파라미터
SGD활용
Learning rate : 0.7, 7.5epochs
128 batch

대부분 문장 20~30개 짧은 문장
미니배치 안 문장들에 대해서는 같은 길이로 맞춰줘서 속도 2배 향상

(3) Experiments

3.5 Parallelization

A C++ implementation of deep LSTM with the configuration from the previous section on a single GPU processes a speed of approximately 1,700 words per second. This was too slow for our purposes, so we parallelized our model using an 8-GPU machine. Each layer of the LSTM was executed on a different GPU and communicated its activations to the next GPU (or layer) as soon as they were computed. Our models have 4 layers of LSTMs, each of which resides on a separate GPU. The remaining 4 GPUs were used to parallelize the softmax, so each GPU was responsible for multiplying by a 1000×20000 matrix. The resulting implementation achieved a speed of 6,300 (both English and French) words per second with a minibatch size of 128. Training took about a ten days with this implementation.

3.5 병렬화

단일 GPU에선 느려서
8-GPU 병렬화

미니배치 크기 128
초당 6300 단어
10일 걸림

(3) Experiments

3.6 Experimental Results

Method	test BLEU score (ntst14)
Baseline System [29]	33.30
Cho et al. [5]	34.54
State of the art [9]	**37.0**
Rescoring the baseline 1000-best with a single forward LSTM	35.61
Rescoring the baseline 1000-best with a single reversed LSTM	35.85
Rescoring the baseline 1000-best with an ensemble of 5 reversed LSTMs	**36.5**
Oracle Rescoring of the Baseline 1000-best lists	~45

Table 2: Methods that use neural networks together with an SMT system on the WMT'14 English to French test set (ntst14).

a sizeable margin, despite its inability to handle out-of-vocabulary words. The LSTM is within 0.5 BLEU points of the previous state of the art by rescoring the 1000-best list of the baseline system.

3.6 결과

BLEU 점수로 번역 품질 평가

제일 좋은 결과보다는 낮지만
유사한 결과
0.5차이

(3) Experiments

3.7 Performance on long sentences

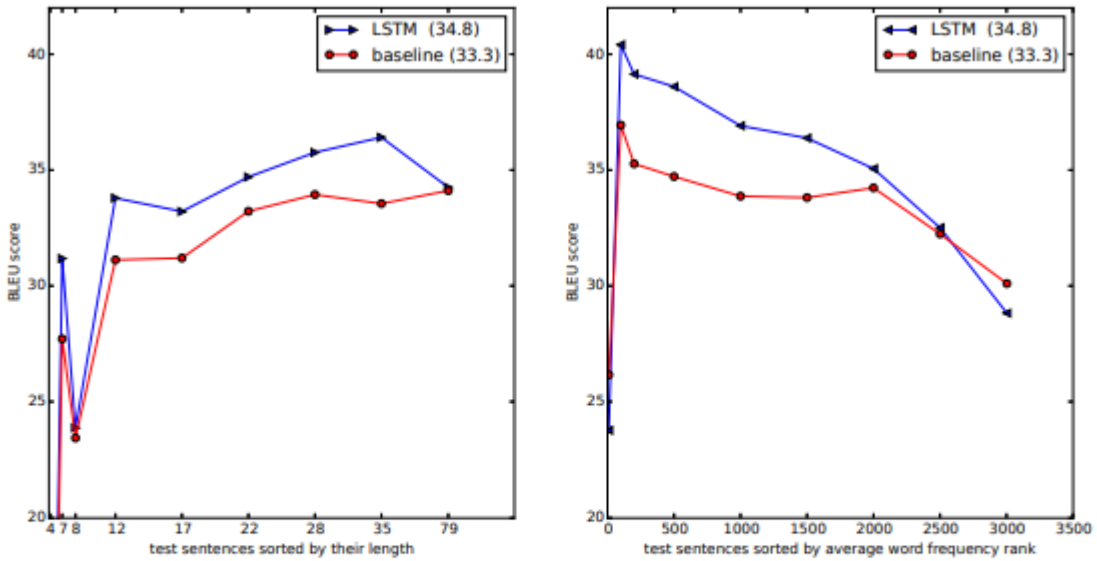
We were surprised to discover that the LSTM did well on long sentences, which is shown quantitatively in figure 3. Table 3 presents several examples of long sentences and their translations.

3.7 긴 문장에서 성능

긴 문장에 대해 잘 작동한다.
가장 긴 문장에서만 약간의 저하만 존재

Type	Sentence
Our model	Ulrich UNK , membre du conseil d' administration du constructeur automobile Audi , affirme qu' il s' agit d' une pratique courante depuis des années pour que les téléphones portables puissent être collectés avant les réunions du conseil d' administration afin qu' ils ne soient pas utilisés comme appareils d' écoute à distance .
Truth	Ulrich Hackenberg , membre du conseil d' administration du constructeur automobile Audi , déclare que la collecte des téléphones portables avant les réunions du conseil , afin qu' ils ne puissent pas être utilisés comme appareils d' écoute à distance , est une pratique courante depuis des années .
Our model	" Les téléphones cellulaires , qui sont vraiment une question , non seulement parce qu' ils pourraient potentiellement causer des interférences avec les appareils de navigation , mais nous savons , selon la FCC , qu' ils pourraient interférer avec les tours de téléphone cellulaire lorsqu' ils sont dans l' air " , dit UNK .
Truth	" Les téléphones portables sont véritablement un problème , non seulement parce qu' ils pourraient éventuellement créer des interférences avec les instruments de navigation , mais parce que nous savons , d' après la FCC , qu' ils pourraient perturber les antennes-relais de téléphonie mobile s' ils sont utilisés à bord " , a déclaré Rosenker .
Our model	Avec la crémation , il y a un " sentiment de violence contre le corps d' un être cher " , qui sera " réduit à une pile de cendres " en très peu de temps au lieu d' un processus de décomposition " qui accompagnera les étapes du deuil " .
Truth	Il y a , avec la crémation , " une violence faite au corps aimé " , qui va être " réduit à un tas de cendres " en très peu de temps , et non après un processus de décomposition , qui " accompagnerait les phases du deuil " .

Table 3: A few examples of long translations produced by the LSTM alongside the ground truth translations. The reader can verify that the translations are sensible using Google translate.



문장길이별 성능 / 희귀한 단어 포함된 문장의 성능

Figure 3: The left plot shows the performance of our system as a function of sentence length, where the x-axis corresponds to the test sentences sorted by their length and is marked by the actual sequence lengths. There is no degradation on sentences with less than 35 words, there is only a minor degradation on the longest sentences. The right plot shows the LSTM's performance on sentences with progressively more rare words, where the x-axis corresponds to the test sentences sorted by their "average word frequency rank".

(3) Experiments

3.8 Model Analysis

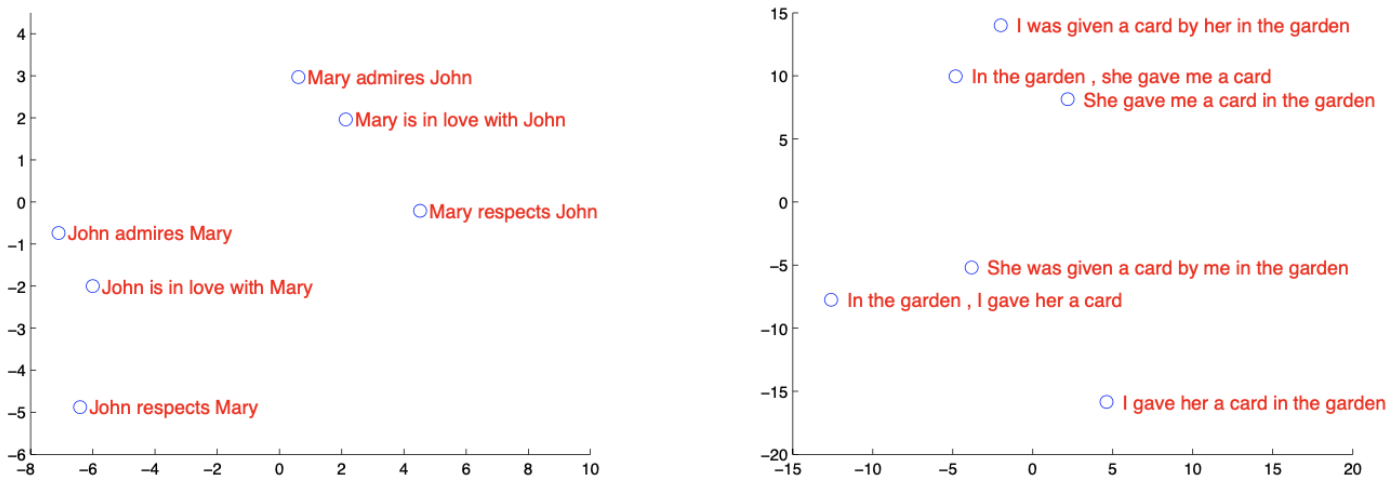


Figure 2: The figure shows a 2-dimensional PCA projection of the LSTM hidden states that are obtained after processing the phrases in the figures. The phrases are clustered by meaning, which in these examples is primarily a function of word order, which would be difficult to capture with a bag-of-words model. Notice that both clusters have similar internal structure.

One of the attractive features of our model is its ability to turn a sequence of words into a vector of fixed dimensionality. Figure 2 visualizes some of the learned representations. The figure clearly shows that the representations are sensitive to the order of words, while being fairly insensitive to the replacement of an active voice with a passive voice. The two-dimensional projections are obtained using PCA.

3.8

최종 번역 결과물에 대해 2차원 주성분 분석 결과

단어의 순서에 있어서는 매우 민감하나

문장의 능동 수동 관계에는 덜 민감한 것으로 확인됨.

(4) Related work

There is a large body of work on applications of neural networks to machine translation. So far, the simplest and most effective way of applying an RNN-Language Model (RNNLM) [23] or a Feedforward Neural Network Language Model (NNLM) [3] to an MT task is by rescoring the n-best lists of a strong MT baseline [22], which reliably improves translation quality.

More recently, researchers have begun to look into ways of including information about the source language into the NNLM. Examples of this work include Auli et al. [1], who combine an NNLM with a topic model of the input sentence, which improves rescoring performance. Devlin et al. [8] followed a similar approach, but they incorporated their NNLM into the decoder of an MT system and used the decoder's alignment information to provide the NNLM with the most useful words in the input sentence. Their approach was highly successful and it achieved large improvements over their baseline.

Our work is closely related to Kalchbrenner and Blunsom [18], who were the first to map the input sentence into a vector and then back to a sentence, although they map sentences to vectors using convolutional neural networks, which lose the ordering of the words. Similarly to this work, Cho et al. [5] used an LSTM-like RNN architecture to map sentences into vectors and back, although their primary focus was on integrating their neural network into an SMT system. Bahdanau et al. [2] also attempted direct translations with a neural network that used an attention mechanism to overcome the poor performance on long sentences experienced by Cho et al. [5] and achieved encouraging results. Likewise, Pouget-Abadie et al. [26] attempted to address the memory problem of Cho et al. [5] by translating pieces of the source sentence in way that produces smooth translations, which is similar to a phrase-based approach. We suspect that they could achieve similar improvements by simply training their networks on reversed source sentences.

End-to-end training is also the focus of Hermann et al. [12], whose model represents the inputs and outputs by feedforward networks, and map them to similar points in space. However, their approach cannot generate translations directly: to get a translation, they need to do a look up for closest vector in the pre-computed database of sentences, or to rescore a sentence.

신경망을 기계번역에 적용하는 작업들

- 최근에, 소스언어에 대한 정보를 NNLM에 포함하는 방법을 조사하기 시작
- Auli
- 입력 문장의 주제 모델과 결합하여 재채점 성능을 향상시킴 -> 성공적
- 우리의 작업은 Ka~, Blun~과 밀접한 관련이 있음
- 입력문장을 처음으로 벡터에 매핑한 다음 다시 문장으로 매핑했지만, 단어 순서를 잃어버리는 컨볼루션 신경망을 사용해 문장을 벡터에 매핑함.
- 긴 문장의 성능을 극복하기 위해 attention을 사용한 신경망으로 직접 번역도 시도, 고무적인 결과를 얻었고, 메모리 문제를 해결하려고 시도도 해봄.구문기반 접근방식과 유사한 원활한 번역을 생성하는 방식으로 원본 문장의 일부를 번역함.뒤집힌 소스 문장에 대한 네트워크를 단순히 훈련함으로써 유사한 개선을 달성할 수 있다고 생각함.
- End-to-end 훈련은 피드 포워드 네트워크에 의한 입력과 출력을 나타내며 공간의유사한 지점에 매핑한다. 그러나 번역을 직접 생성하지 못하고, 미리 계산된 문장 DB에서 가장 가까운 벡터를 찾아 문장을 다시 채점해야함.

(5) Conclusion

In this work, we showed that a large deep LSTM with a limited vocabulary can outperform a standard SMT-based system whose vocabulary is unlimited on a large-scale MT task. The success of our simple LSTM-based approach on MT suggests that it should do well on many other sequence learning problems, provided they have enough training data.

We were surprised by the extent of the improvement obtained by reversing the words in the source sentences. We conclude that it is important to find a problem encoding that has the greatest number of short term dependencies, as they make the learning problem much simpler. In particular, while we were unable to train a standard RNN on the non-reversed translation problem (shown in fig. 1), we believe that a standard RNN should be easily trainable when the source sentences are reversed (although we did not verify it experimentally).

We were also surprised by the ability of the LSTM to correctly translate very long sentences. We were initially convinced that the LSTM would fail on long sentences due to its limited memory, and other researchers reported poor performance on long sentences with a model similar to ours [5, 2, 26]. And yet, LSTMs trained on the reversed dataset had little difficulty translating long sentences.

Most importantly, we demonstrated that a simple, straightforward and a relatively unoptimized approach can outperform a mature SMT system, so further work will likely lead to even greater translation accuracies. These results suggest that our approach will likely do well on other challenging sequence to sequence problems.

이 작업에서, 제한된 어휘를 가진 대규모 deep LSTM이 대규모 MT작업에서 어휘가 무제한인 표준 SMT기반 시스템을 능가할 수 있음을 보여주었다.

소스 문장의 단어를 뒤집음으로써 얻은 개선의 정도에 놀람. 우리는 학습 문제를 더 단순하게 만들기 위해 단기 종속성을 갖는 문제의 종속성을 찾는 것이 중요하다고 보았다.

또한 매우 긴 문장을 정확하게 번역하는 LSTM의 능력에 놀람.

가장 중요한건, 간단한 상대적으로 최적화되지 않은 접근방식이 성숙한 SMT 시스템을 능가할 수 있음을 입증했기 때문에 추가 작업을 통해 번역 정확도를 훨씬 더 높을 수 있을 것으로 기대한다.