Masterarbeit im Fach Informatik
Rheinisch-Westfälische Technische Hochschule Aachen
Lehrstuhl für Informatik 6
Prof. Dr.-Ing. H. Ney

Unsupervised Learning of Neural Network Lexicon and Cross-lingual Word Embedding

20. September 2018

vorgelegt von:
Autor Jiahui Geng
Matrikelnummer 365655

Gutachter:
Prof. Dr.-Ing. H. Ney
Prof. B. Leibe, Ph. D.

Betreuer:
M.Sc. Yunsu Kim

Erklärung

Geng, Jiahui
Name, Vorname

365655
Matrikelnummer (freiwillige Angabe)

Ich versichere hiermit an Eides Statt, dass ich die vorliegende ~~Arbeit~~/Bachelorarbeit/~~Masterarbeit~~* mit dem Titel

Unsupervised Learning of Neural Network Lexicon and Cross-lingual Word Embedding

selbständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt. Für den Fall, dass die Arbeit zusätzlich auf einem Datenträger eingereicht wird, erkläre ich, dass die schriftliche und die elektronische Form vollständig übereinstimmen. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

Aachen, 20. September 2018
Ort, Datum

Unterschrift

*Nichtzutreffendes bitte streichen

Belehrung:

§ 156 StGB: Falsche Versicherung an Eides Statt

Wer vor einer zur Abnahme einer Versicherung an Eides Statt zuständigen Behörde eine solche Versicherung falsch abgibt oder unter Berufung auf eine solche Versicherung falsch aussagt, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.

§ 161 StGB: Fahrlässiger Falscheid; fahrlässige falsche Versicherung an Eides Statt

(1) Wenn eine der in den §§ 154 bis 156 bezeichneten Handlungen aus Fahrlässigkeit begangen worden ist, so tritt Freiheitsstrafe bis zu einem Jahr oder Geldstrafe ein.

(2) Straflosigkeit tritt ein, wenn der Täter die falsche Angabe rechtzeitig berichtigt. Die Vorschriften des § 158 Abs. 2 und 3 gelten entsprechend.

Die vorstehende Belehrung habe ich zur Kenntnis genommen:

Aachen, 20. September 2018
Ort. Datum

Unterschrift

Abstract

In recent years, cross-lingual representation of words enable us to explore the structures of different languages, based on that similarity we can realize bilingual dictionary induction and information retrieval, it also enable the knowledge transfer bwtween languages especially between resource-rich and resource-lean languages.
In this thesis we first make a survey of the cross-lingual word embedding training methods and its application in unsupervised machine translation. Then we proposed a novel training method of cross-lingual word embedding, which is data-driven and supported with language model, we can better utilize the context information for training. we further exploit the cross-lingual similarity in word-by-word based unsupervised machine translation combined with context-aware beam search and denoising autoencoder.
The performance of the cross-lingual embedding is measuared on an open dictionary from Facebook, we find the performance is comparable with the state-of-the-art supervised and unsupervised methods. Based on that, our simple yet efficient unsupervised machine translation system can produce meaningful machine translation the BLEU scores even get beyond some unsupervised system with costly iterative training.
We also make some ablation studies, analyze the effects of various factors in the cross-lingual word embedding training and various artificial noise. We think our work can provide better understanding of the training and the applications of word embedding in MT.

Contents

Chapter 1

Introduction

Building a good machine translation system requires a huge collections of parallel data. NMT systems often fail when the training data is not enough. The lack of parallel corpora is actually a problem. Parallel corpora are costly because they require huge human labor, time and expertise of corresponding languages. Several approaches has been proposed to alleviate this issue, for instance, triangulation and semi-supervised learning techniques, however these systems still need a strong cross-lingual signal. Unsupervised machine translation uses only monolingual data of the source and target language to train the model and such monolingual corpora is readily available.

Recently cross-lingual word embedding draws more and more attention since it helps to exploit the multilingual semantic information, furthermore enables computation of cross-lingual word similarities, which is relevant for many tasks such as bilingual dictionary induction or cross-lingual information retrieval. Several methods are proposed to learn the cross-lingual without any parallel data. This inspires the works on unsupervised machine translation as We can make use of such cross-lingual information in an unsupervised manner.

In this thesis, I first make a survey of the current training algorithm of cross-lingual word embedding, find the significant training skills among them and their respective advantages and disadvantages. Then I propose a novel data-based unsupervised training method, the experiments the method can achieve considerable results in comparison with the start-of-the-art method. I further explore the application of the cross-lingual word embeddings in the machine translation domain. I develop a total unsupervised machine translation system starting from the simple word-by-word translation principle, to over the shortcomings to such more, I design the context-aware beam search to find the best translation candidate. To handle to reordering, I implement a denoising network with artificial noises, which are to mimic the true noise in the word-by-word translation, The results demonstrate that such simple but efficient translation system can surpass the most unsupervised neural machine translation system with costly iterative training.

1.1 Related Work

1.1.1 Word Embedding

Traditional language processing systems treat words as discrete atomic symbols: they assign for each word a specific id number. Such encodings are arbitrary, and it does not provide any information about the relations that may exist between individual symbols. They are at their limits in many tasks. In comparison, the distributed representations of words in high dimensional continuous space help learning algorithm to achieve better performance in natural language processing. According to the distributional hypothesis, similar words tends to occurs with similar neighbors, so similar words have similar word representations. At the beginning, in order to explore word similarity, those training algorithms involve dense matrix multiplication and complex matrix analysis.
Mikolov et al. [2013a] propose two models, namely skip-gram model and continuous bag of words model (CBOW). They used neural architectures for learning word vectors, which makes training extremely efficient. In further work, Mikolov et al. [2013c] discuss about learning the embedding for phrases. Pennington et al. [2014] propose a regression model, which combines the global matrix factorization and local context window methods, make the training process more interpretable. Bojanowski et al. [2016] represent each word as a bag of character n-grams, and assign for each character n-gram a distinct embedding. In this way, we can exploit the subword information.
Mikolov et al. [2013b] again notice that continuous embedding spaces exhibit similar structures across languages, even when for distant language pairs, for example, English-Vietnamese. With separately learned word embeddings he is committed to exploiting semantic word similarities, concretely by learning a linear mapping from source embedding space to target embedding space. They employ a parallel vocabulary of five thousand words as anchor points to learn this mapping and evaluate their approach on a word translation task. Xing et al. [2015] show that results can be improved by enforce an orthogonal constraint on the linear mapping.
Recently, Artetxe et al. [2017a] raise an iterative method that align the word embedding spaces gradually. Thanks to his contribution, the vocabulary size for learning can be reduced to 25 word pairs and even with Arabic numerals. The performance is comparable to the learning with large vocabulary dictionary Cao et al. [2016] propose a distribution-based model, which is a modified CBOW model which tries to minimization between the distribution dissimilarity between source and target embeddings. Here distribution information refers the mean and variance. Zhang et al. [2017] put forward a method using adversarial training without any parallel data. The discriminator tried to discriminate if the embedding from the source side and the generator aimed to learning the mapping from source embedding space to

target one. These methods are totally unsupervised but the performances are far worse than the supervised training. Conneau et al. [2017] simplifies the adversarial training structure with different loss functions for generator and discriminator. The performance got much improved, They also proposed to use cross-domain similarity local scaling (CSLS) to handle hubness and a validation criterion for unsupervised model selection.

1.1.2 Unsupervised Machine Translation

As mentioned previously, the lack of parallel corpora motivate people to use monolingual data to improve the machine translation system. Some researchers tried to use triangulation techniques (Cohn and Lapata [2007]) and semi-supervised approaches (Cheng et al. [2016]) to alleviate this issue. But these methods still require parallel corpora.
As to fully unsupervised machine translation, which only takes advantages of monolingual data, Ravi and Knight [2011] first consider it as a deciphering problem, where the source language is considered as ciphertext. They also proposed iterative expectation-maximization method (EM) and Bayesian decipherment to train this unsupervised model. To solve the bottleneck of such model mainly the huge memory required to store the candidates search space, Nuhn et al. [2012] limit search candidates according to the word similarity. Nuhn and Ney [2014] limit the search space by using beam search and preselection search. Kim et al. [2017] enforce the sparsity with a simple threshold for the lexicon. He also initialize the lexicon training with word classes, which efficiently boosts the performance. Although initially not based on distributional semantics, recent studies show that the use of word embeddings can bring significant improvement in statistical decipherment Duong et al. [2016].
He et al. [2016] raise dual learning for neural machine translation. More concretely, they train two agents to translate in opposite directions (e.g. French $\rightarrow$ English and English $\rightarrow$ French) and make them teach each other through a reinforcement learning process. While promising, this approach still requires a small parallel corpora for a warm start. Artetxe et al. [2017b] and Lample et al. [2017] propose two bi-directional unsupervised machine translation model which totally rely only on monolingual corpora in each language. These two models both need to use cross-lingual word embedding to initialize the MT system and train the sequence-to-sequence system with denoising autoencoder. They turn the unsupervised training into a supervised one by introducing back-translation techniques. The most important principle for these two works are the shared embedding space. Sentences from different languages are encoded into a shared embedding spaces and then translated into specific language with different decoder.

1.2 Outline

The remainder of this thesis is structured as follows. In Section 1.3 we introduce the notations in this thesis. Chapter 2 introduces the development of machine translation systems from statistical models to neural models. Some basic techniques and principles for machine translation are also included. Chapter 3 gives more information, I do a survey of training and applications details of cross-lingual word embedding. We discuss the unsupervised learning of cross-lingual word embedding and our efforts on data-driven training. We describe our unsupervised machine translation model in chapter 5, which contains mainly context-aware method with the help of language model and denoising autoencoder method aimed at reordering. We demonstrate the results of cross-lingual word embedding model and unsupervised machine model. We will compare the performance with the state-of-the-art model. In Chapter 7 we summarize our work.

1.3 Notation

In this thesis, we use the following notations:

- source sentence $f_1^J := f_1 \cdots f_j \cdots f_J$
- target sentence $e_1^I := e_1 \cdots, e_i \cdots e_I$
- single character e, f
- $\boldsymbol{f}$ and $\boldsymbol{e}$ the source and target word embedding
- $\boldsymbol{E}$, $\boldsymbol{F}$ are the corresponding embedding matrices
- source and target Corpora $\mathcal{F}, \mathcal{E}$
- $P(e_1^I|e_1^J)$ and $P(f_1^J|f_1^I)$ for denoising autoencoder
- $P(e_1^I|f_1^J)$ and $P(f_1^J|e_1^I)$ for translation model
- s and s_i internal states of RNN decoder
- h and h_i internal states of RNN encoder
- c and c_i for context vectors
- α: the alignment

Chapter 2

Machine Translation

This chapter describes the classical or start-of-the-art machine translation models from statistical machine translation model to neural machine model. The relative research works on unsupervised machine translation will also be included.

2.1 Statistical Machine Translation

Statistical machine translation has achieved success until the beginning of this century. The initial statistical models for machine translation are based on words as atomic units that may be translated, inserted, dropped and reordered. In statistical machine translation, we use both a translation model and a language model which ensures fluent output. Later the statistical machine translation prefers to use translation of phrases as atomic units. These phrases are any contiguous sequences of words, not necessarily linguistic entities. In this approach, the input sentence is broken up into a sequence of phrases, these phrases are mapped one-to-one to output phrases, which may be reordered.

2.1.1 Word-Based Model

Noisy-channel model: Noisy channel model based on the notion of a noisy channel from Shannon's information theory has been applied to many language processing problems. Assume the source sentence is a distorted message omitted from the target sentence, we have a model on how the message is distorted (translation model $Pr(f_1^J|e_1^I)$) and also a model on which original messages are probable (language model $Pr(e_1^I)$), our task is to find the best translation e_1^I for an input foreign sentence.

$$\begin{aligned}\underset{e_1^I}{\operatorname{argmax}}\{Pr(e_1^I|f_1^J)\} &= \underset{e_1^I}{\operatorname{argmax}}\frac{Pr(f_1^J|e_1^I)Pr(e_1^I)}{Pr(f_1^J)} \\ &= \underset{e_1^I}{\operatorname{argmax}}\{Pr(f_1^J|e_1^I)\cdot Pr(e_1^I)\}\end{aligned}$$

The basic model for word-based model is noisy-channel model. The task of word alignment is an artifact of word-based machine translation. Alignment models target at the reordering problem for word-based translation.

IBM Models

Based on the noisy-channel model, IBM word-based translation make the model more complicated by adding submodels. Starting from lexical translation, absolute alignment model, fertility etc. are added step by step.

Alignment model introduce an explicit model for reordering words in a sentence. More often than not, words that follow each other in one language have translations that follow each other in one language have translations that follow each other in the output language. In detail, the position j in the source sentence is aligned with the position i in the target sentence when translating, denoted as $i = a_j$. Alignment model is a global reordering model.For whole sentence, we denote the alignment as

$$a_1^J := a_1 \cdots a_J$$

Fertility model the specific number of output words in the output language. Generally one word in one language just translate into one single word in the other language. But some words produce multiple words or get dropped. we denote the fertility model as $\phi(e)$, so the length of translation sentence

$$J = \sum_{i=0}^{I} \phi(e_i)$$

The word-based translation process can be described as the following figure, we use IBM-3 model for example: IBM-3 model contains four steps:

- Fertility step
- NULL insertion
- Lexical translation step
- Distortion step for reordering

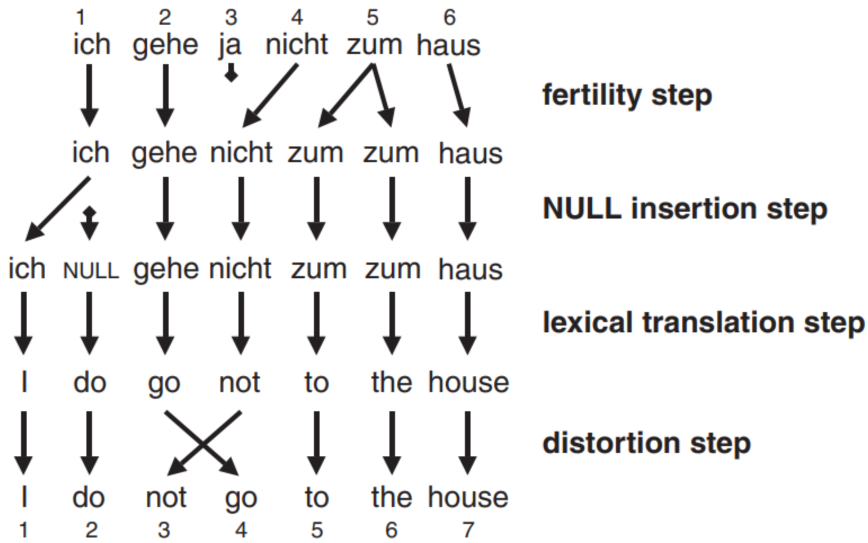

Figure 2.1: Illustration of IBM-3 model (Koehn [2009])

Weighted Model

To overcome shortcomings in modeling, introduce scaling exponents λ_m as in speech recognition. For example

$$Q(f_1^J, e_1^I; a_1^J) = P(J|I)^{\lambda_1} \cdot \prod_i P(e_i|e_h)^{\lambda_2} \cdot \prod_j [p(a_j|a_{j-1}, I, J)^{\lambda_3} \cdot p(f_j|e_{a_j})^{\lambda_4}]$$

where the four probabilities corresponds to length model, language model, alignment model and lexical model separately. e_h denotes the history words in N-gram model

2.1.2 Phrase-Based Model

Actually when translating, words may not be the best candidates for the smallest units for translation, sometimes one word in a foreign languages should be translated into two English words, or vice versa. Word-based models often break down in these cases.
Phrase-based models typically do not strictly follow the noisy-channel approach proposed for word-based models, but use a log-linear framework This model allow s straightforward integration if additional features.

Log-linear Model Combination: Consider arbitrary models ("feature functions"):

$$q_m(f_1^J, e_1^I; a_1^J) > 0 \quad m = 1, \cdots, M$$

$$\begin{aligned} Q(e_1^I, f_1^J; a_1^J) &= \prod_{m=1}^{M} q_m(f_1^J, e_1^I; a_1^J)^{\lambda_m} \\ &= \exp(\sum_{m=1}^{M} \lambda_m \log q_m(f_1^J, e_1^I; a_1^J)) \end{aligned}$$

In this frame work, we view each data point as a vector of features and the model as a set of corresponding feature functions, these functions are trained separately and combined assuming that they are independent of each other.

Components for log-linear model can be such as language model, phrase translation model, reordering model are used as feature functions with appropriate weights.

- Phrase translation model can be learned from a word-aligned parallel corpus, alternatively we also can use expectation maximization algorithm to directly find phrase alignment for sentence pairs.
- Reordering model for phrase case is typically modeled by a distance based reordering cost that discourage reordering in general. Lexicalized reordering model can be introduced for specific phrase pair.

The model can be further extended by components like: bidirectional translation probabilities, lexical weighting, word penalty and phrase penalty.

2.1.3 Decipherment

Ravi and Knight [2011] frame the MT problem as a decipherment task, treating the foreign text as a cipher. They also propose iterative expectation-maximization method (EM) and Bayesian decipherment to train this unsupervised model. Many concepts in deciperment are the same as the SMT.

IBM model tries to maximize the probability with hidden alignment model

$$\underset{\theta}{\operatorname{argmax}} \prod_{\boldsymbol{e},\boldsymbol{f}} P_\theta(\boldsymbol{f}|\boldsymbol{e}) = \underset{\theta}{\operatorname{argmax}} \prod_{\boldsymbol{e},\boldsymbol{f}} \sum_{a} P_\theta(\boldsymbol{f}, a|\boldsymbol{e})$$

While for unsupervised case, we train

$$\underset{\theta}{\operatorname{argmax}} \prod_{\boldsymbol{f}} P_\theta(\boldsymbol{f}) = \underset{\theta}{\operatorname{argmax}} \prod_{\boldsymbol{f}} \sum_{\boldsymbol{e}} P(\boldsymbol{e}) \cdot P_\theta(\boldsymbol{f}|\boldsymbol{e})$$

for hidden alignments:

$$\underset{\theta}{\text{argmax}} \prod_{\boldsymbol{f}} \sum_{\boldsymbol{e}} P(\boldsymbol{e}) \cdot \sum_{a} P_{\theta}(\boldsymbol{e}, a|\boldsymbol{e})$$

Since the model is very complicated, more assumptions are added to the model. The model accounts for word substitutions, insertions, deletions and local reordering during the translation process but does not incorporate fertilities or global re-ordering.

The generative process:

1. Generate an English sentence $\boldsymbol{e}$ with probability $P(\boldsymbol{e})$.
2. Insert a NULL word at any position in the English sentence with uniform probability.
3. For each English word token e_i (including NULLs), choose a foreign word translation f_i, with probability $P_{\theta}(f_i|e_i)$, the foreign word may be NULL.
4. Swap any pair of adjacent foreign words f_{i-1}, f_i, with probability $P_{\theta}(swap)$.
5. Output the foreign string f_1^M, skipping over NULLs.

However, this method is limited to rather short sentences and simplistic settings.

2.2 Neural Machine Translation

Neural machine translation (NMT) has recently become the dominant method for machine translation task. In comparison to the traditional statistical machine translation (SMT), NMT systems are trained end-to-end, taking advantages of the continuous representation of the hidden states that greatly alleviate the sparsity problem and make better use of more contextual information. Besides this, traditional phrase-based machine translation, which consists of several models that are tuned separately, neural machine translation tries to build a more general neural network model which can directly output translations given input, it contains only one single model, and only one single training criterion.
Sutskever et al. [2014] first use a multi-layer Long Short-Term Memory (LSTM) to to train the machine translation system, attention mechanism has lately been used to improve neural machine translation by selectively focusing on parts of the source sentence during translation. The inherently sequential nature precludes parallelization within training examples. Convolutinoal sequence to sequence (ConvSeq2seq) and Transformer architectures are brought forward for significantly more parallelization during training to better exploit the GPU hardware and these models can reach a new state of the art in translation quality.

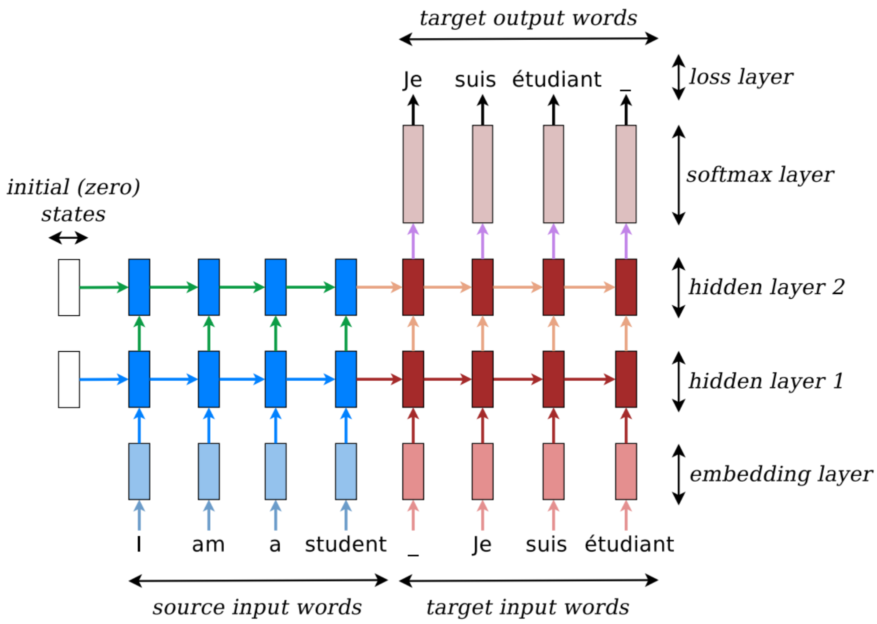

Figure 2.2: Neural machine translation – example of a deep recurrent architecture (Luong et al. [2015a])

2.2.1 Encoder-Decoder Framework

The most common network structure is the encoder-decoder framework, in which two recurrent neural networks work together to transform one sequence to another. An encoder condenses an input sequence into a vector $\boldsymbol{c}$, and a decoder unfolds that vector into a new sequence. The model can be expressed as:

$$p(e_1^I|f_1^J) = \prod_i p(e_i|e_0^{i-1}, f_1^J) = \prod_i p(e_i|e_0^{i-1}, \boldsymbol{c})$$

The RNN decoder predicts the next word on the basis of the internal states and the previous word. In more detail, one can parameterize the probability of decoding each word e_i as:

$$p(e_i|e_0^{i-1}, \boldsymbol{c}) = \text{softmax}(g(\boldsymbol{s_i}))$$

with g being the transformation function that outputs a vocabulary-size vector. Here $\boldsymbol{s}_i$ is the RNN hidden state, updated as:

$$\boldsymbol{s}_i = f(\boldsymbol{s}_{i-1}, \boldsymbol{c})$$

It is common in neural machine translation systems to use a beam-search to sample the probabilities for the words in the sequence output by the model. The wider the beam width, the more exhaustive the search, and, it is believed, the better the results.

Drawbacks:

1. The model compresses all information from the input sentence into a hidden vector 1, while ignores the length of input sentence, when the length of input sentence get very long, even longer than the training sentences, it becomes harder to extract specific information for predicting the target word, the performance will get worse.

2. It's not suitable to assign the same weight to all input words, one target word corresponds usually to one or several words in the input sentence. Treating all words equally does not distinguish the source information and influence the performance badly.

2.2.2 Attention Mechanism

Alignment is the problem in machine translation that identifies which parts of the input sequence are relevant to each word in the output, whereas translation is the process of using the relevant information to select the appropriate output. Bahdanau et al. [2014] introduce an extension to the encoder-decoder model which learns to alignment and translate jointly. Each time the model predicts the next target word, it softly searches for a set of positions in a source sentence where the most relevant information is concentrated. The model then predicts a target word based on the context vectors associated with these source positions and all the previous generated target words.
Described in formula, attention mechanism derives a context vector $\boldsymbol{c}_i$ that capture the input information to help to predict the target word at the position i. Given the target hidden state $\boldsymbol{s}_i$ and the source-side context vector $\boldsymbol{c}_i$, we can compute the hidden state $\tilde{\boldsymbol{s}}_i$ by combining the current hidden state $\boldsymbol{s}_i$ and the context vector $\boldsymbol{c}_i$:

$$\tilde{\boldsymbol{s}}_i = tanh(W_c[\boldsymbol{c}_i; \boldsymbol{s}_i])$$

Then the target word is correspondingly predicted by softmax function:

$$p(e_i|e_0^{i-1}, f_1^J) = softmax(W_s\tilde{\boldsymbol{s}}_i)$$

Global attention & Local attention
Global attention attends all the input words, weighted average of source hidden states (word representations):

$$\boldsymbol{c}_i = \sum_j \boldsymbol{\alpha}(j|i) \cdot \boldsymbol{h}_j$$

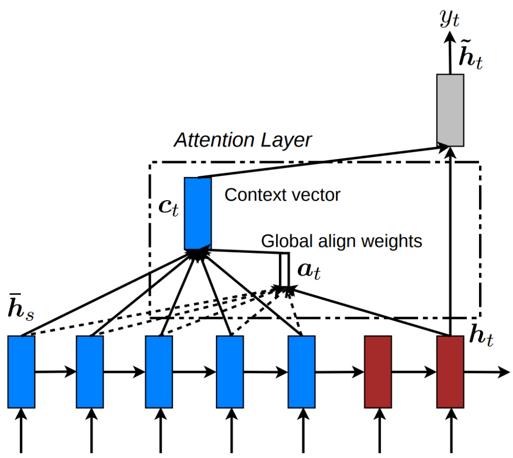

Figure 2.3: Global attention

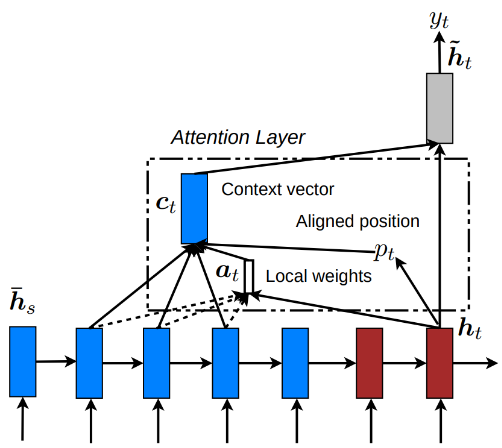

Figure 2.4: Local attention

where $\boldsymbol{\alpha}(j|i)$ is the normalized attention weight that output at position i is aligned with input position j, and it can be calculated as the following softmax-like function:

$$\boldsymbol{\alpha}(j|i) = \text{align}(\boldsymbol{s}_i, \boldsymbol{h}_j) \tag{2.1}$$

$$= \frac{\exp\ (\text{score}(\boldsymbol{s}_i, \boldsymbol{h}_j))}{\sum_{\boldsymbol{j'}} \exp\ (\text{score}(\boldsymbol{s}_i, \boldsymbol{h}'_j))} \tag{2.2}$$

Since score function is referred as a content based function, different forms can be considered:

$$\text{score}(\boldsymbol{s_i}, \boldsymbol{h}_j) = \begin{cases} \boldsymbol{s_i}^T \boldsymbol{h}_j & \textit{dot} \\ \boldsymbol{s_i}^T W_a \boldsymbol{h}_j & \textit{general} \\ \boldsymbol{v_i}^T \textit{tanh}(W_a[\boldsymbol{s_i}; \boldsymbol{h}_j]) & \textit{concat} \end{cases} \tag{2.3}$$

For global attention, for each target word we need to attend the whole input sentence, it is very expensive and impractical to translate longer sentences. Luong et al. [2015a]) proposed the local attention, the local attention is actually the trade-off between soft and hard attention.

Local attention first predicts an aligned position $a(i)$ for each target word at position i. Then the context vector $\boldsymbol{c}_i$ is a weighted sum within the window $[a(i) - D, a(i) + D]$, D is selected empirically. The model predict the aligned position p_i as followed:

$$a(i) = S \cdot \text{sigmoid}(\boldsymbol{v}_p^T \tanh(W_p \boldsymbol{s}_i))$$

W_p and $\boldsymbol{v}_p$ are the model parameters which will be learned to predict positions. S is the source sentence length. As a result of sigmoid function, $a(i) \in [0, S]$. To favor

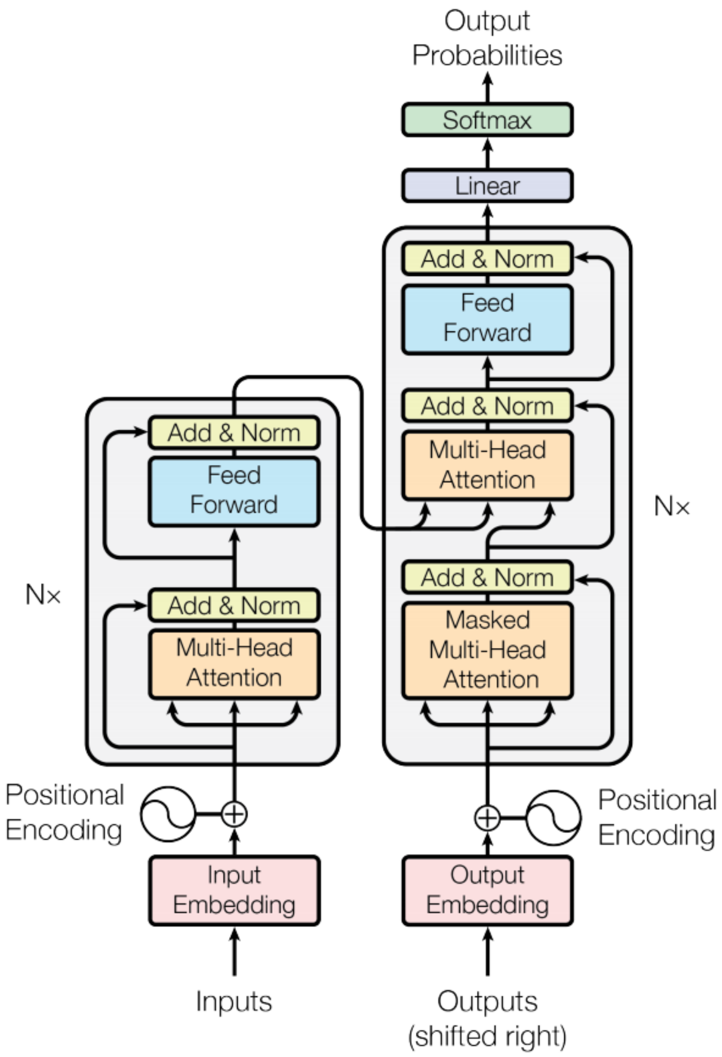

Figure 2.5: The Transformer model architecture (Vaswani et al. [2017])

the words near position $a(i)$, they place a Gaussian distribution which centered at $a(i)$.

$$\boldsymbol{\alpha}_i(j) = \text{align}(\boldsymbol{s}_i, \boldsymbol{h}_j) \cdot \exp\Big(-\frac{(j - a(i))^2}{2\sigma^2}\Big)$$

2.2.3 Transformer

The RNN encoder-decoder models have achieved state of the art in sequence modeling and machine translation problem. However such RNN models also have some disadvantages, because of their inherently sequential computation which prevents parallelization across elements of the input sequence. That means it is more difficult to fully take advantage of the modern computing devices like GPU and TPU. Convolutional Gehring et al. [2017] and fully-attentional feed-forward architectures like Transformer Vaswani et al. [2017] models are proposed as alternatives for RNNs.

Dot-Product Attention Given a query $\boldsymbol{q}$ and a set of key-value $(\boldsymbol{k}-\boldsymbol{v})$ pairs, the output is weighted average of values, where the weight of each value is computed by inner product of query and corresponding key. The queries and keys are of the same dimension d_k, values are of dimension d_v. Actually, for better understanding, the query $\boldsymbol{q}$ can be considered as the decoder state $\boldsymbol{s}_i$ and key $\boldsymbol{k}$ and values $\boldsymbol{v}$ can be considered as the encoder state $\boldsymbol{h}_j$ in previous attention model. Then the attention of query $\boldsymbol{q}$ on all $(\boldsymbol{k}-\boldsymbol{v})$ pairs are:

$$A(\boldsymbol{q}, K, V) = \sum_i \frac{\exp\ (\boldsymbol{q} \cdot \boldsymbol{k}_i)}{\sum_j \exp\ \ \boldsymbol{q} \cdot \boldsymbol{k}_j} \cdot \boldsymbol{v}_i$$

When we stack the queries $\boldsymbol{q}$ to Q:

$$A(Q, K, V) = \mathrm{softmax}(QK^T)V$$

As d_k get larger, the variance of $q^T k$ get larger. The softmax become very peaked and the gradient get smaller. To counteract this effect, we scaled the dot products by $\frac{1}{\sqrt{d_k}}$, we get

$$\mathit{Attention}(Q, K, V) = \mathrm{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

The shape of resulting matrix is $n_q \times d_v$

Multi-head attention
First map Q, K, V into h many lower dimension spaces via linear mapping matrix. Here h is the number of heads. Then apply attention and concatenate the outputs. Suppose the original dimensions of queries, keys, values are d_m. We the number of heads is h, then we set $d_k = d_v = d_m/h$. With $i \in 1 \cdots h$ different matrix $W_i^Q \in R^{d_m \times d_k}$ $W_i^K \in R^{d_m \times d_k}$ $W_i^V \in R^{d_m \times d_k}$ and $W^O \in R^{d_m \times d_k}$.

$$\mathrm{MultiHead}(Q, K, V) = Concat(h\mathrm{head}_1, \cdots, \mathrm{head}_h)W^O$$
$$\mathrm{head}_i = \mathit{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

where $\text{head}_i \in \mathbb{R}^{n_q \times d_k}$, $\text{Multihead}(Q, K, V) \in \mathbb{R}^{n_q \times d_m}$

There are three attention types in the transformer model:

- encoder-decoder attention
  the queries come from the previous decoder layer, the keys and values are just the output of decoder. This works as the attention mechanism in the seq2seq model, it allow the decoder to align all positions in the input sequence with different weights

- encoder self-attention
  all queries, keys, values come from the encoder layer, each position allows to attend to all positions before that position

- decoder self-attention
  similar to encoder attention, each position is allowed to attend to position before and including that position

Positional Encoding
Since there is no RNN or CNN structures in transformer model, we need also need to make use of sequence information for seq2seq learning. We need inject position information into the embedding.

$$PE_{(\text{pos},2i)} = sin(\text{pos}/10000^{2i/d_m})$$

$$PE_{(\text{pos},2i+1)} = cos(\text{pos}/10000^{2i/d_m})$$

where pos is the position and i is the dimension. That is, each dimension of the positional encoding corresponding to a sinusoid.

2.3 Neural Unsupervised Machine Translation

give the first results for training a full translation model from non-parallel data, and use this model to translate previously-unseen text. The proposed problem named "deciperment" has received a lot of attention. This model contains many statistical methods, like EM algorithm or Bayesian method. However, this method is limited to rather short sentences and simplistic setting. With the development of neural machine translation, the end-to-end method performs better and easier to tune without much specific linguistic knowledge. The Idea is similar to dual learning framework except that in dual learning model, gradients are backpropagated through the reverse model and pretrain using a relative large parallel data as a warm start. The neural model mapping the sentences from monolingual corpora into the same latent space, By learning to reconstruct in both languages from the shared feature space. The translation model in both direction will be improved synchronously with the back-translation.

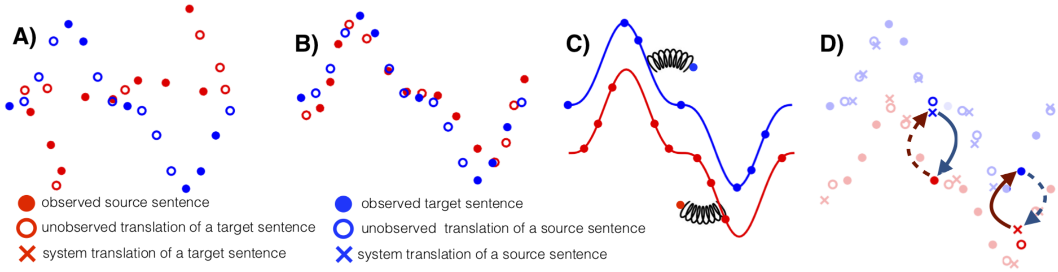

Figure 2.6: Illustration of unsupervised machine translation: A) the two original monolingual data; B) Initialization, the two distribution are roughly aligned; C) Denoising autoencoder, make the distribution closer to normal one in corresponding language D) Back-translation, improve the both translation model in iterative way (Lample et al. [2018])

Dual Structure

Dual structure is inspired by the observation: any machine translation has a dual task: e.g. German-to-English translation (primal) and English-to-German translation (dual). Dual tasks can form a closed loop and generate informative feedback signals to for both translation models. Based on the feedback signals generated during this process, and leverage language model, we can minimize the reconstruction error of the original sentences. We can iteratively update the two models until convergence. He et al. [2016] trained two agents to translate in opposite directions (e.g. French → English and English → French), and make them teach each other through a reinforcement learning process. While promising, this approach still requires a parallel corpus for a warm start. Lample et al. [2018] simplified the structure, in the proposed model, the gradients will not be back-propagated through the reverse model, but only make use of the back-translation. The training process as followed:

Algorithm 1 Unsupervised Machine Translation

Language models: Learn language models P_s, P_t over source and target languages;

Initial translation model: Leveraging P_s, P_t, lean two initial translation models in each direction: $P^{(0)}_{s\to t}$, $P^{(0)}_{t\to s}$;

for k = *1 to N* **do**

  Back-translation: Generate source and target sentences using the current translation models $P^{(k-1)}_{t\to s}$ and $P^{(k-1)}_{s\to t}$, leverading P_s, P_t;

  Retrain: Train new translation models $P^{(k)}_{t\to s}$ and $P^{(k)}_{s\to t}$ using the generated sentences and leverading P_s, P_t;

end

Initialization

If without enough information to start the dual machine translation system, it will be hard for the system to catch the meaningful signals and then it will take much more iterations. However if we initialize the system by a naive a word-by-word translation of the sentence, where the bilingual lexicon are derived from the same monolingual data. While such initial "word-by-word" translation maybe poor if languages or corpora are not closely related, it still preserves some of the original semantics. Such word-by-word translation can already achieve several BLEU scores.

Shared Latent Representation

A shared encoder representation actis like an interlingua, which is translated in the decoder corresponding language regardless less of the input source language. Since the only supervision information only comes from the monolingual data. The model learn to translate by learning to reconstruct in both language from this shared space.

There are two different methods to learn the shared feature space:

- Shared encoder The system use only one encoder that is shared by both languages involved. The universal encoder is aimed to produce a language independent representation of the input language but the decoder should separately translate then into corresponding language.
- Adversarial training Train discriminator to classify between the encoding of source and target sentences. The discriminator operates on the output of the encoder, the encoder is trained instead to fool the discriminator.

Optimization

When minimizing the loss function, gradients will not be back propagated through the reverse model which generate the data. Instead the objective function minimized at every iteration is the sum of L^{auto} and L^{back}

- Denoising autoencoder loss

$$L^{auto} = \mathbb{E}_{\boldsymbol{e}\sim E}[-\log P_{t\to t}(\boldsymbol{e}|\text{noise}(\boldsymbol{e})] + \mathbb{E}_{\boldsymbol{f}\sim F}[-\log P_{s\to s}(\boldsymbol{f}|\text{noise}(\boldsymbol{f}))]$$

  where s $P_{s\to s}$ and $P_{t\to t}$ are the composition of encoder and decoder both operating in the source and target sides, respectively

- Back-translation loss

$$L^{back} = \mathbb{E}_{\boldsymbol{e}\sim T}[-\log P_{s\to t}(\boldsymbol{e}|u(\boldsymbol{e}))] + \mathbb{E}_{\boldsymbol{f}\sim S}[-\log P_{t\to s}(\boldsymbol{f}|v(\boldsymbol{f}))]$$

  we denote the sentence that translated by intermediate target-to-source translation model as $u(y)$, similarly denote the sentence translated by source-to-target model as $v(x)$, so $u(y)$ should in source language and $v(x)$ should in target language. The pairs $(x, v(x))$, $(u(y), y)$ constitute synthetic parallel sentences.

Chapter 3

Word Embedding

Word embeddings is distributed representation of words in a vector space. With the learning algorithm it can capture the contextual or co-occurrence information. The word embedding has an interesting and important property: similar words will have similar distribution in the embedding space, with that property, we can find meaningful near-synonyms or Some successful methods for learning word embedding like word2vecMikolov et al. [2013c], Pennington et al. [2014]

3.1 Monolingual Embedding

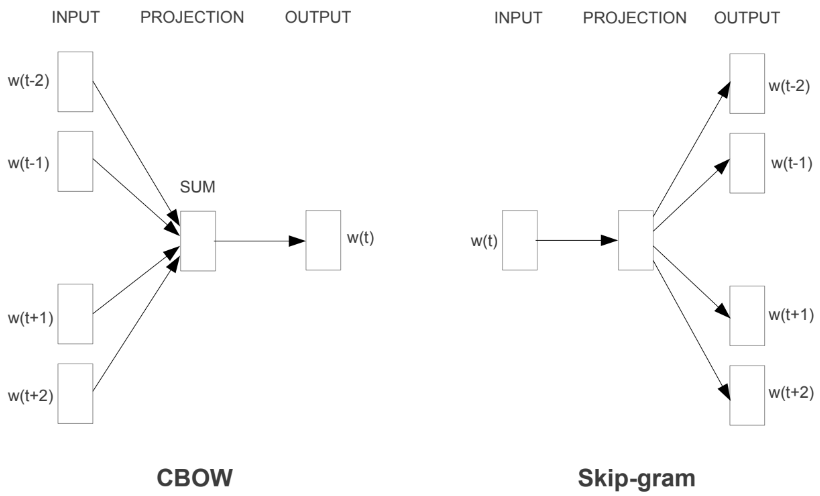

Figure 3.1: Global attention model (Mikolov et al. [2013a])

3.1.1 CBOW and Skip-gram Model

CBOW model and skip-gram model are currently mainstream neural network to learn the word embedding. Algorithmically, CBOW tries to predict the current word based on the context while skip-gram model tries to maximize classification of a word based on another word in the same sentence. Using skip-gram model as example, given a large training corpus represented as a sequence of words $w_1, \cdots w_N$, the objective of the model is to maximize the following log-likelihood:

$$\sum_{s=1}^{N} \sum_{w_t \in \mathcal{C}_s} \log \ p(w_t|w_s)$$

where w_s is the current word, the context $\mathcal{C}^s$ is the set of words surrounding word w_s, so w_t means the word to predict. So probability above can be interpreted as observing a context word w_t given w_s.

Suppose we can given a scoring function s which maps word pairs to score in $\mathbb{R}0$ like cosine similarity, the probability can be defined like softmax function:

$$p(w_t|w_s) = \text{softmax}(s(w_t, w_s)) \tag{3.1}$$

$$= \frac{\exp\{s(w_t, w_s)\}}{\sum_{w' \in W} \exp\{s(w', w_s)\}} \tag{3.2}$$

where W is the whole vocabulary.

We train the model by maximizing its log-likelihood on the training dataset, i.e. by maximizing:

$$J_{ML} = \log p(w_t|w_s) \tag{3.3}$$

$$= s(w_t, w_s) - \log(\sum_{w' \in W} \exp\{s(w', w_s)\}) \tag{3.4}$$

Noise-Contrastive Training
However the normalization on the whole vocabulary is very expensive since it is conducted for all words at every training step. The problem of predicting words can be considered as an independent binary classification task. For example in the skip-gram model, we consider all the context words as positive samples and the words randomly sampled from the dictionary as the negative ones. Then the training objective is

$$J_{NEG} = \log Q_\theta(D = 1|w_t, w_s) + \sum_{w' \sim P_{noise}} \log Q_\theta(D = 0|w', w_s)$$

where $Q_\theta(D = 1|w_t, w_s)$ is the binary logistic regression probability. In practice, we draw k contrastive words from the noise distribution. Then the model are instead trained to discriminate the read target word w_t, from imaginary (noise) words.

Since we only calculate the loss function for k samples instead the whole vocabulary, it becomes much faster to train.
According to empirical results, CBOW works better on smaller datasets because CBOW smoothes over a lot of the distributional information while Skip-Gram model performs better when we have larger datasets

3.1.2 FastText

The training methods above treat each word as a distinct word embedding, however intuitively we can obtain more information from the morphological information of words. A subword model was proposed to try to fix such problem.The training network is similar, the model design a new presentation of the word: it adds speicial symbols $<, >$ as boundary information at the beginning and the end of a word. Then a normal word is represented as a bag of character n-grams . For example the word "where" and n equals 3, the it can be represented as the following 5 tri-grams:

$$< wh, whe, her, ere, re >$$

Suppose in this way we denote a word w as G_w the set of character n-grams, we assign for each character n-gram g in G_w, we assign a distinct vector z_g, we will finally represent the embedding of word w as the sum of these vector and also for the scoring function:

$$s(w, w_s) = \sum_{g \in G_w} z_g^T w_s$$

3.2 Cross-lingual Word Embedding

Cross-lingual word embedding is defined as word embedding of multiple languages in a joint embedding space. Mikolov first notice that the embedding distributions exhibit similar structure across languages. They proposed to use a linear mapping from the source embedding to target embedding.

Cross-lingual word embeddings are appealing due to two main factors: they enable not only cross-lingual semantics, reasoning about word meaning in multilingual contexts, for example, we can induce the bilingual dictionary by calculating the cross-lingual word similarities, but also knowledge transfer between languages, most importantly between rich-resource and lean-resource languages (Adams et al. [2017]).
In the thesis, I assume there are two set of embeddings e, $f$trained separately on monolingual data. The propose of cross-lingual word embedding training is to learn such a mapping $W \in$ from source embedding space to target embedding space, so

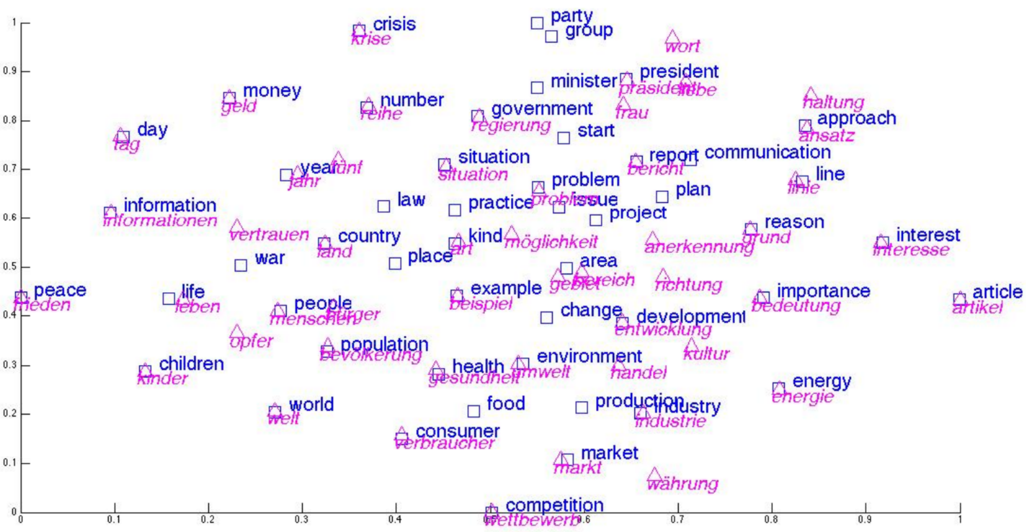

Figure 3.2: A cross-lingual embedding space between German and English (Ruder et al. [2017])

Wf_i, e_j in the same embedding space and for all corresponding word pairs, we need to optimize the mapping W, so that"

$$\arg \min_{W \in R^{d \times d}} \sum_i \|Wf_i - e_i\|$$

where d is the dimension of embeddings, and the distance $\|Wf_i - e_i\|$ can be different types. We prefer the Euclidean distance.
first observe that word embeddings trained separately on monolingual corpora exhibits isomorphic structure across languages, as illustrated in Figure . That means we can create a connection between source embedding and target embedding even with simple linear mapping. This has far-reaching implication on low-resource scenarios , because word embedding requires only plain text to train, which is the most abundant form of linguistic resource.

3.2.1 Training Algorithm

Most training methods can be divided into joint training approaches and mapping-based approaches. , . first train monolingual word representations on corresponding corpus separately. Then mappings are learned to project different embeddings into the shared space. They learn the transformation from word alignments or bilingual dictionaries. Some works focus on learning the mapping from the source embed-

ding space to the target embedding space. The criterion is to minimize the mean squared error between the transformed source embeddings and the target embeddings. Another type like CCA-based mapping use canonical correlation analysis (CCA) to project words from two languages into a shared embedding space ().
Joint Training Approaches
Joint training means to optimize the source and target embeddings objectives $\mathcal{L}(\cdot)$ jointly with the cross-lingual regularization term $\Omega(\cdot)$.

$$\mathcal{L} = \mathcal{L}^{e}_{\text{skip-gram}} + \mathcal{L}^{f}_{\text{skip-gram}} + \lambda \cdot \Omega(\cdot)$$

where the first two losses are exact the same as those in monolingual embedding training. While the $\Omega(\cdot)$ encourages the similar words representations to be similar for words that are related across different languages.
Different cross-lingual regularization terms are proposed to optimize the cross-lingual embeddings (Coulmance et al. [2016], Luong et al. [2015b], Gouws et al. [2015]). For example, Gouws et al. [2015] models it as:

$$\Omega_{\text{BilBOWA}} = \| \frac{1}{I} \sum_{e_i \in e_1^I} \boldsymbol{e}_i - \frac{1}{J} \sum_{f_j \in f_1^J} \boldsymbol{f}_j \|^2$$

where $\boldsymbol{f}_j$ and $\boldsymbol{e}_i$ are the word embeddings for words in parallel source and target sentences. Instead of relying on expensive word alignments which minimize the distance that aligned to each other, they minimize the distance between the mean of the word representations in the aligned sentences.
Mapping-based Approaches:
First train monolingual word embeddings independently on large monolingual corpora and then find the most suitable transformation matrix that maps the embeddings into a shared embedding space. The mapping is trained based on bilingual dictionary.
Multilingual learning can be categorized into mapping-based approaches and regularization-based approaches. In the mapping-based approaches, the embedding is performed for each language individually with monolingual data, and then the mapping are learned using multilingual data to represent the relation between the languages. One method is to learn the linear mapping projection, another is to learn mappings that project word vectors of all languages to a common low-dimension space, where the correlation of the multilingual word pairs is maximized with the canonical correlation analysis (CCA). The advantages of approach is that it is very fast to learn the embedding alignments. The main draw back of this approach is that it is not clear if a single transformation whether linear or nonlinear can capture the relationship between all words in different languages.
The regularization-based approaches involve the multilingual constraint in the objective function for learning the embedding, adds an extra term that reflects the

distances of some pairs of semantically related words from different languages into the objective function.

Canonical Correlation Analysis (CCA)

let $E \in \mathbb{R}^{m \times d_1}$ and $F \in \mathbb{R}^{n \times d_2}$ be word embeddings of two different vocabularies where rows represent words. Since the two vocabularies are of different sizes and there might be not exist translation of every word. Let $\boldsymbol{e}$,$\boldsymbol{f}$ be two corresponding vectors from , and be the two projection directions. Then the projected vectors are:

and the correlation between the projected vectors can be written as:

$$\rho(\boldsymbol{f}', \boldsymbol{e}') = \frac{\mathbb{E}[(\mho)()]}{\sqrt{\mathbb{E}[(\mho)^{\nvDash}]\sqrt{\boldsymbol{E(e)^2}}}}$$

$$\frac{\boldsymbol{u}^T \Sigma_{fe} \boldsymbol{v}}{\sqrt{\boldsymbol{u}^T \Sigma_{ff} \boldsymbol{u}}\sqrt{\boldsymbol{v}^T \Sigma_{ee} \boldsymbol{v}}}$$

where $\Sigma_P fe$ and Σ_{ff} are the cross-view and within
CCA maximizes ρ for the given set of vectors and and outputs two projection matrix :
Using these two prokection vectors we can project the entire vocabularies of the two languages
Deep Canonical Correlation Analysis A linear feature mapping is often not sufficiently powerful to faithfully capture the hidden, non-linear relations within the data. proposed a non-linear extension of CCA using deep neural networks. In this model, two neural networks are used to extract features from each voew, and trained to maximize the correlations between outputs in the two views, measured by a linear CCA steo with projection mappings the neural network weights and linear projections are optimized together using the objective

The joint formulation of the learning objective provided as:

$$J = \mathcal{L}^s + \mathcal{L}^t + \Omega(s, t)$$

where $\mathbb{L}^s$ and $\mathbb{L}^t$ are monolingual objectives in each language optimized jointly with the cross-lingual regularization objective.

exact cross-lingual objective as follows:

$$\Omega(E,F) = \sum_i \sum_j a_{ij} |\boldsymbol{e}_i - \boldsymbol{f}_j|^2$$
$$= (E-F)^2 A (E-F)$$

where subscript A indicates that the alignments are fixed. A captures the relationships between all source and target vocabularies.
A dictionary is necessary for learning the cross-lingual word embedding. minimizing the distance in a bilingual dictionary.

Xing showed that the results are improved when we constrain the W to be an orthogonal matrix. This constraint, the optimal transformation can be efficiently calculated in linear time with respect to the vocabulary size.
The problem then is simplified as the Procrustes problem and there exists a closed-form solution obtained from the SVD of EF^T
Orthogonal Constraints Starting from Mikolov et al. [2013b], the mapping from source embedding space to target embedding space can be represented as a linear repression. The objective can be defined as:

$$\min_W \sum_i |Wf - e\|^2$$

Since we retrieve the word translation according to cosine similarity, it's better to solve the problem by redefine the optimization function using the cosine distance:

$$\operatorname*{argmax}_W \sum_i (Wf_i)^T e^i$$

. We consider the source and target embedding in the same space. In this case, the normalization constraint on word vectors can be satisfied by constraining W as an orthogonal matrix. This is equivalent to minimizing the (squared) Frobenius norm of the residual matrix:

$$W^* = \operatorname*{argmin}_W \|WF - E\|_F^2$$

The problem boils down to the Procrustes problem which has a closed form solution obtained from the singular value decomposition (SVD).

$$W* == UV^T, \quad U\Sigma V^T = SVD(EF^T)$$

CSLS Loss Inspired from the work of Conneau et al. [2017], where the dictionary inducted from CSLS loss: $\boldsymbol{e}$

$$CSLS(\boldsymbol{e},\boldsymbol{f}) = -2cos(\boldsymbol{e},\boldsymbol{f}) + \frac{1}{k}\sum_{\boldsymbol{e}'\in N_{\boldsymbol{e}}(\boldsymbol{f})} cos(\boldsymbol{e}',\boldsymbol{f}) + \frac{1}{k}\sum_{\boldsymbol{f}'\in N_{\boldsymbol{f}}(\boldsymbol{e})} cos(\boldsymbol{f}',\boldsymbol{e})$$

since we have $cos(\boldsymbol{W}\boldsymbol{e}, \boldsymbol{f}) = \boldsymbol{e}^T\boldsymbol{W}^T\boldsymbol{f}$
The loss function can be rewritten as:

$$\min_{\boldsymbol{W}\in} = \frac{1}{n}\sum_{i=1}^{n}$$

Minimization of a non-smooth cost function over the manifold of orthogonal matrices . Instead of using manifold optimization tools, **?** proposed to derive convex relaxations that can lead to a simple and tractable minimization algorithm.

- Spectral norm: replacing the set of orthogonal matrices 1 by its convex hull, that is the set of matrices with singular values smaller than 1, the unit ball of the spectral norm
- Frobenius norm: replacing the the set of orthogonal matrices 1 with ball of radium $\sqrt{d}$ in Frobenius norm

With such two relaxations, the CSLS loss is constrained to a convex function with respect to the mapping $\boldsymbol{W}$.
We train the linear mapping with in the spectral norm by projected gradient descent:2 For each iteration, train the mapping $\boldsymbol{W}$ with gradient descent, then constrain the mapping by projection of the set.

- Spectral norm
  take the SVD of the trained matrix, threshold the singular values to one
- Probenius norm
  divide the matrix by its Frobenius norm

Chapter 4

Cross-lingual Word Embedding without Parallel Data

Proposed cross-lingual embedding learning requires several thousands of word pairs as anchors to learn the mapping with good generalization ability to predict for unseen word. But as for low-source language, dictionary of good quality is not readily available. This motivates more and more works on unsupervised cross-lingual word embedding learning, if the unsupervised method works, the word-level or grammar-level properties will be better captured, since it enables the knowledge transfer for those less-studied languages.

4.1 Initialization

Vulic and Korhonen [2016] shows that a seed dictionary of at least hundreds are need for the model to generalize. Hoshen and Wolf [2018] proposed an approach which first use PCA to realize an approximate alignment, then use mini-batch cycle iterative closest point to learn the mapping. Zhang et al. [2017] proposed an adversarial autoencoder, using the generator G to implement the mapping so that the transformed source embedding similar to the corresponding target embedding, and the discriminator D strives to distinguish the target embedding from the transformed source embedding. Conneau et al. [2017] simplifies the structure and proposed an unsupervised criterion that is highly correlated with the quality of the mapping which can be used as stop criterion and to select the best hyper-parameters.

4.1.1 Iterative Closest Point Method

The method assumes that many language pairs share some principle axes of variation. For each language, first select most frequent word vector,center the data and project it to the top p principle components. The normal Iterative Closest Point (ICP) is unidirectional, the Mini-batch Cycle ICP (MBC-ICP) includes cycle-constrains ensuring that a word transformed to another language could also be transformed back and stay unchanged. Each iteration the final MBC-ICP algorithm can described as:

1. For each target word e, find the nearest $W_{s\to t}f(e)$, denote the source word $f(e)$
2. For each source word f, find the nearest $W_{t\to s}e(f)$, denote the target word $e(f)$
3. Optimize $W_{e\to f}$ and $W_{f\to e}$:

$$\begin{aligned}\mathcal{L} = &\sum_e \|e - W_{s\to t}f(e)\| + \sum_f \|f - W_{t\to s}e(f)\| \\ &+ \lambda \sum_e \|e - W_{s\to t}W_{t\to s}e\| + \lambda \sum_f \|f - W_{t\to s}W_{s\to t}f\|\end{aligned}$$

4.1.2 Adversarial training

Generative adversarial networks are a class of artificial intelligence algorithm used in unsupervised learning. The objective of generative network (generator) is to increase the error rate of the discriminative network (discriminator) while the discriminator discriminates between instances from the true data distribution and candidates produced by the generator. Following this principle, the generator tries to learning to mapping from the source embedding space to target embedding space, so that the discriminator while could be a deep neural network cannot distinguish the data source.
Let $= \{x_1, \cdots, x_n\}$ and $= \{y_i, \cdots, y_m\}$ be the two sets of n and m word embeddings from a source and a target language separately, We refer the discriminator parameters as θ_D.The discriminator is a multi-layer neural network trained to discriminate the transformed source word embedding from the target word embedding, while the mapping W, simply a linear transformation, is trained to fooling discriminator. In the two-player game, we are supposed to learn the mapping from source embedding space to the target space.
In the first adversarial training method for cross-lingual word embedding, the objective of the two networks are as followed:
Discriminator objective

$$L_D(\theta_D|W) = -\frac{1}{n}\sum_{i=1}^{n} \log P_{\theta_D}(source = 1|Wf_i) - \frac{1}{m}\sum_{i=1}^{m} \log P_{\theta_D}(source = 0|e_i)$$

Generator objective

$$L_W(W|\theta_D) = -\frac{1}{n}\sum_{i=1}^{n} \log P_{\theta_D}(source = 0|Wf_i)$$

To relax the orthogonal constraint, the authors introduce the adversarial autoencoder, after the mapping $W$from source embedding space to target embedding space, the source embedding should also be mapping back with W^T, therefore, they introduce the reconstruction loss as

$$L_W(W|\theta_D) = -\frac{1}{n}\sum_{i=1}^{n}\{\log P_{\theta_D}(source = 0|Wf_i) - \lambda\cos(f_i, W^TWf_i)\}$$

In Conneau et al. [2017] work, they just take a symmetric loss for the generator loss and achieve better results:

$$L_W(W|\theta_D) = -\frac{1}{n}\sum_{i=1}^{n}\log P_{\theta_D}(source = 0|Wf_i) - \frac{1}{m}\sum_{i=1}^{m}\log P_{\theta_D}(source = 1|e_i)$$

Model Selection

Since the cross-lingual embedding training is under the unsupervised setting, We do not know the word translation accuracy, otherwise if we have the validation data, that means we will have parallel data, against the unsupervised idea. To address this issue, we must select from the property of data or the loss of the neural network as the unsupervised criterion. However in the experiments we find that the accuracy of the discriminator always stays at a high level no matter how is the word translation accuracy.
All these methods can be use to find meaningful word pairs in both languages. For further refinement we can use the induced dictionary to start the iterative self-learning algorithm.

4.2 Nearest Neighbor Search

Hubness Problem
Points are tending to be nearest neighbors of many points in high-dimensional space. Since we use nearest neighbor search, those points (hubs) will harm the search accuracy.

Inverted Softmax
The confidence of choosing a target word as translation of a source word can be considered as softmax-like normalized probability

$$p(e|f) = \frac{\exp\Big(\beta \cdot s(e, f)\Big)}{\sum_{e'}\exp\Big(\beta \cdot s(e', f)\Big)}$$

where the $s(\cdot)$ is the score function we can define ourselves. we learn the "temperature" β by maximizing the log probability over the training dictionary.

$$\underset{\beta}{\operatorname{argmax}} \sum_{e,f} \ln p(e|f)$$

The author observed that, if we invert the softmax and normalizing the probability over all the source words rather than target words, the hubness problem could be mitigated:

$$p(e|f) = \frac{\exp\Big(\beta \cdot s(e,f)\Big)}{\alpha\ \sum_{f'} \exp\Big(\beta \cdot s(e,f')\Big)}$$

Cross-domain Similarity Local Scaling
We denote $N(f)$ the set of K nearest neighbors of points of the mapped f in the target embedding space, and $N(e)$ the nearest neighbors of mapped e in the source embedding space. We consider the mean cosine similarity as hub-ness:

$$r(f) = \frac{1}{K} \sum_{e' \in N(f)} \cos(e', Wf)$$

So in this way we penalize the hub points:

$$CSLS(\boldsymbol{e}, \boldsymbol{f}) = 2cos(\boldsymbol{e}, \boldsymbol{y}) - \frac{1}{K} \sum_{\boldsymbol{e'} \in N(\boldsymbol{e})} cos(\boldsymbol{f}, \boldsymbol{e'}) - \frac{1}{K} \sum_{\boldsymbol{f'} \in N(\boldsymbol{e})} cos(\boldsymbol{f'}, \boldsymbol{e})$$

With the nearest neighbor search we can evaluate the performance of learned mapping, by checking if the nearest neighbors of the source word are also in the candidates in a open word translation dictionary dataset.

4.3 Iterative Training

Assume that with the initialization step we can already get roughly aligned distribution of the source and target embedding, and we can use this mapping to learn a good dictionary (word pairs). Since the quality of dictionary gets improved, we can further learn a better mapping, consequently, such process can be repeated iteratively until convergence criterion is met.

Algorithm 2 Iterative training procedure

Input: $\mathcal{F}$ (source embeddings)
Input: $\mathcal{E}$ (target embeddings)
Input: $\mathcal{D}$ (seed dictionary)
Result: $\mathcal{W}$ (embedding mapping)
while *not converge* **do**
  $\mathcal{W} \leftarrow LEARN\_MAPPING(\mathcal{F}, \mathcal{E}, \mathcal{D})$
  $\mathcal{D} \leftarrow LEARN\_DICTIONARY$
end

4.3.1 Self-learning with Procrustes

For self-learning frame work, we compute the optimal dictionary based on the mapped source word embeddings and target word embeddings. To sure the quality of the small dictionary, we choose only top frequent words for both languages and only bidirectional translations are kept. We even filter the dictionary with some threshold.

4.3.2 Data-driven Method

Since we have large monolingual data and one of the tasks to evaluate the cross-lingual word embedding is build machhine translation system based on these cross-lingual embeddings. We make use of the language model and improve also the translation quality simultaneously. We can set the source from the translation, build the dictionary from the word translation pairs, in detail the training procedure are as followed:

1. translate corpus according to current mapping, get the word pairs D
2. train the network with D to minimize the mapping distance
3. Repeat $1, 2$ until the algorithm converges

$$\left.\begin{array}{cc} (f_1, & e_1) \\ (f_2, & e_2) \\ \vdots & \\ (f_N, & e_N) \end{array}\right\} \Rightarrow D$$

Previously, the objective we want to minimize is the mean square error of the mapped embeddings. By training the mapping with neural network we can define different loss functions according to specific features

$$\mathcal{L} = \sum_{(f,e)\in D} \|Wf - e\|^2$$

Chapter 5

Sentence Translation

As mentioned previously, training a traditional machine translation system requires large parallel data. With cross-lingual word embedding we can already found ambiguous word translation, in this chapter we propose a simple yet effective method to improve quality of translation which starts from the word-by-word translation. We integrate monolingual model like language model and denoising neural network of the target side to produce meaningful sentence translation. Since all the sub-models are trained on monolingual corpora, our proposed method is total unsupervised. Our system surpasses state-of-the-art unsupervised translation without costly iteratively training.

5.1 Context-aware Beam Search

5.1.1 Language Model

Language models are widely applied in natural language processing, especially machine translation which need frequent queries.
n-gram models
N-gram language models use the Markov assumption to break the probability of a sentence into the product of the probability of each word given a limit history of preceding words.

$$p(w_1^N) = \prod_{i=1}^{N} p(w_i|w_1, \cdots w_{i-1}) = \prod_{i=1}^{N} p(w_i|w_{i-(n-1)}, \cdots, w_{i-1})$$

The conditional probability can be calculated from n-gram model frequent counts:

$$p(w_i|w_{i-(n-1)}, \cdots, w_{i-1}) = \frac{count(w_{i-(n-1)}, \cdots, w_i)}{count(w_{i-(n-1)}, \cdots, w_{i-1})}$$

Language model tries to handling sparse data problem because some words or phrases have not been seen yet in the training corpus does not mean they are not impossible. Different smoothing techniques like back-off or interpolation are implemented to assign a probability mass to unseen cases.

5.1.2 Beam Search

The complexity of a search graph is exponential to the length of the given source sentence. Beam search is a heuristic search algorithm that explores a graph by expanding the most promising nodes. At each step of the search process, it will evaluate all the candidates together with the reserved translation results from last step, it will only stores a predetermined number (beam width) of translations for next step. The greater the beam width is, the fewer states will be pruned. So it is suggested to prune these word translation candidates as soon as possible to reduce the search space and speed up the translation. According to the similarity of cross-lingual word embedding, we are able to find some meaningful for translation candidates for a given word. But there are also words that actually noise in the candidates or obviously incorrect because of grammar checking. With the support of language model, we can select the most probable words from previous word translation candidates.
Given a history h of target word before e, the score of e to be the translation of f is defined as:

$$\hat{e}_1^N = \underset{e_1^N}{\operatorname{argmax}} \prod_{n=1}^{N} p^{\lambda_{LM}}(e_n|e_{n-4}^{n-1}) \cdot q^{\lambda_{emb}}(f_n, e_n)$$

where the lexicon score $q(f, e) \in [0, 1]$ defined as:

$$q(f, e) = \frac{d(f, e) + 1}{2}$$

$d(f, e) \in [-1, 1]$ cosine similarity between f and e
In experiments, we find such lexicon score works better than others, e.g. sigmoid or softmax

5.2 Denoising Neural Network

5.2.1 Denoising Auto-encoder

With the help of language model we have actually improved the quality of word-by-word translation but the results are still far from a acceptable one because of the drawback of the word-by-word mechanism, maybe to some degree we can infer the meaning of sentence, but the sequence of sentences depends on the specific language. We implement the sequential denoising autoencoder to improve the translation output.
An autoencoder is a neural network that is trained to copy its input to its output, autoencoders minimize the loss function like:

$$L(\boldsymbol{x}, g(f(\boldsymbol{x})))$$

where L penalizing the difference between the input and output. While a denoising autoencoder (DAE) instead minimizes

$$L(\boldsymbol{x}, g(f(\tilde{\boldsymbol{x}})))$$

where $\tilde{\boldsymbol{x}}$ is a noise transformation of $\boldsymbol{x}$ and denoising autoencoder will try to learn to ignore the noise in $\boldsymbol{x}$ reconstruct the correct one. Sequential denoising autoencoder will find robust representation of sentences. In practice, denoising autoencoder consists of two parts, namely encoder and decoder. The encoder processes noised data and produces real-valued vectors as an encoding feature of the data. The computational graph of the denosing autoencoder, which attempt to reconstruct the normal input $\boldsymbol{x}$ from it corrupted version $\tilde{\boldsymbol{x}}$. The model is trained by minimize the loss.
For our sequential denoising model, the label sequences would be the monolingual data of the target language. However we do not have the noise input. In order to make the model run correctly, we should mimic the noise sentence of word-by-word translation on the target monolingual corpus.
We design different noise types w.r.t. the word-by-word translation. In the experiments, we inject the artificial noise into a clean sentence, the experiment results shows the noise is reasonable and suitable in this case.

5.2.2 Noise Model

We design three types of noise to handle the fertility and reordering problem, namely reordering noise, insertion noise and deletion noise. In experiments, the noise model can improve the sentence translation, but since it actually starts from the word-by-word translation, it can only deal with reordering in limited distance, cannot work for global reordering.

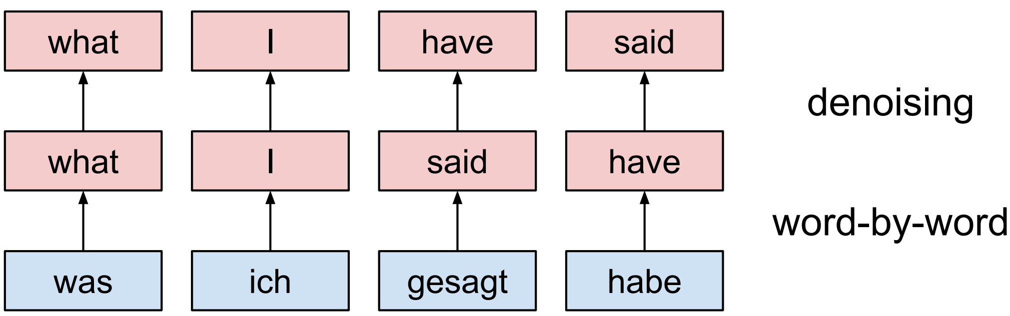

Figure 5.1: Reordering noise

Reordering Noise

The reordering problem is a common phenomenon in the word-by-word translation since the sequence in source language is not exact the sequence in target language. For example, in the grammar of German, the verb is often placed at the end of the clause. "um etwas zu tun". However in English it is not the case, the corresponding translation sequence is "to do something". The verb should always before the noun. In our beam search, language model only assisting in choosing more suitable word from the translation candidates, it cannot reorder the word sequence at all.
For a clean sentence from the target monolingual corpora, we corrupt the word sequence by permutation operation. We limit the maximum distance between the original position and its new position.
The design of reordering noise is as followed:

1. For each position i, sample an integer δ_i from $[0, d_{per}]$
2. Add δ_i to index i and sort $i + \delta_i$
3. Rearrange the words to be in the new positions, to which where indices have been moved

Reordering is actually depends on the specific language pair. However in the experiments we found the performance of the denoising network aimed at such noise is not obvious. The Bleu score before and after the process is close.

Insertion Noise

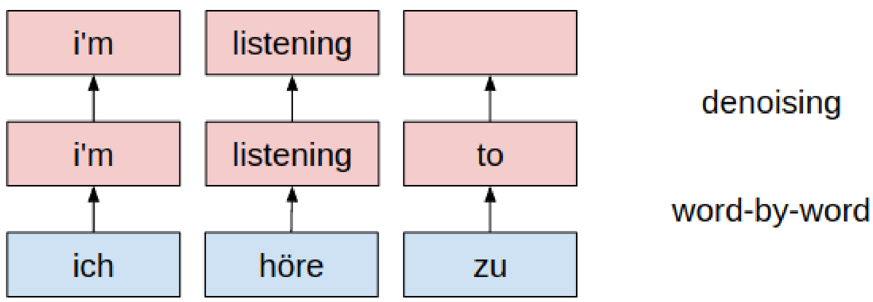

Figure 5.2: Insertion noise

The word-by-word translation system predict the source word at every position of the sentence. However the vocabularies of different systems are not symmetric, for example, in German there are more compound words than that in English. So when translating cross languages, there are a plenty of cases that a single word will be translated to multiple words and multiple words correspond to a single conversely. We focus on such a case: from a German sentence: "ich höre zu" to "i'm listening". A very frequent word "zu" which corresponds to "to" in English, is dropped from the sentence. The design of reordering noise is as followed:

1. For each position i, sample a probability $p_i \sim \text{Uniform}(0, 1)$
2. If $p_i < p_{ins}$, sample a word e from the most frequent V_{ins} target words and insert it before the position$i$

We limit the insertion word in a set consisting of the top frequent word in the target language V_{ins}

Deletion Noise
The deletion noise is just a contrary case of insertion noise. Because we are limited to generate only ne word per source word, it is also possible that a target word in the reference is not related to any source word. For example for "eine der besten" the corresponding translation is "one of the best". We need to add an extra preposition in the target sentence. To simulate such situation, we drop some words randomly from a clean target sentence.

1. For each position i, sample a probability $p_i \sim \text{Uniform}(0, 1)$
2. If $p_i < p_{del}$, drop the word in the position i

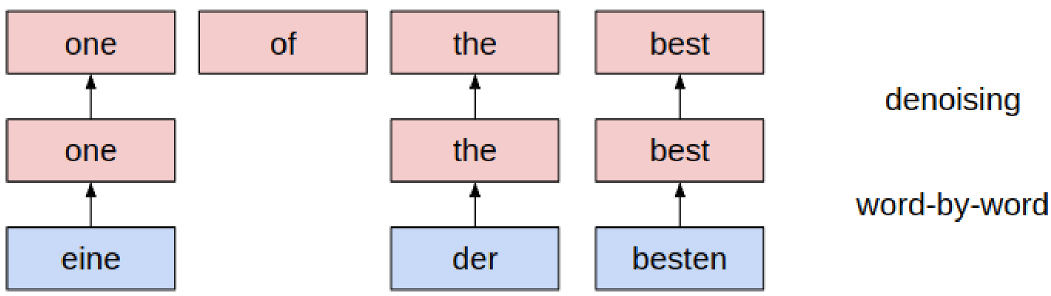

Figure 5.3: Deletion noise

Chapter 6

Experiments

6.1 Corpus Statistics

6.2 title

6.3 title

		German	English	French
Train	**Sentences**	**100M**	**100M**	**100M**
	Running Words	**1880M**	**2360M**	**3017M**
	Vocabulary	**1254k**	**523k**	**660k**

		`newstest2016`		`newstest2014`	
		German	**English**	**French**	**English**
Test	**Sentences**	**2999**	**2999**	**3003**	**3003**
	Running Words	**62506**	**64619**	**81165**	**71290**
	Vocabulary Size	**11978**	**8645**	**10899**	**9200**
	OOV Rates	**4116 (6.6%)**	**1643 (2.5%)**	**1731 (2.1%)**	**1299 (1.8%)**
	LM perplexity	**211.0**	**109.6**	**51.2**	**84.6**

Search vocabulary in testing: 50k (src/tgt)

Table 6.1: Translation results on German↔English `newstest2016` and French↔English `newstest2014`.

System	de-en Bleu [%]	en-de Bleu [%]	fr-en Bleu [%]	en-fr Bleu [%]
Word-by-Word	11.1	6.7	10.6	7.8
+ LM (5-gram) + tgt w/ high LM score for OOV	12.9	8.9	12.7	10.0
+ LM (5-gram) + copy from src for OOV	14.5	9.9	13.6	10.9
+ Denoising (RNN)	16.2	10.6	15.8	13.3
+ Denoising (Transformer)	17.2	11.0	16.5	13.9
Lample et al. [2017]	13.3	9.6	14.3	15.1
Artetxe et al. [2017b]	-	-	15.6	15.1

Table 6.2: Word-by-word translation from German to English

	Accuracy [%]	Bleu [%]
5M	44.9	9.7
10M	51.6	10.1
50M	59.4	10.8
100M	61.2	11.2

Vocabulary		Bleu [%]
	Merges	
BPE	20k	10.4
	50k	12.5
	100k	13.0
	Cross-lingual training	
Word	20k	14.4
	50k	14.4
	100k	14.5
	200k	14.4

d_{per}	p_{del}	p_{ins}	V_{ins}	Bleu [%]
2				14.7
3				14.9
5				14.9
3	0.1			15.7
	0.3			15.1
3	0.1	0.1	10	16.8
			50	17.2
			500	16.8
			5000	16.5

Vocabulary			No LM Bleu [%]	With LM Bleu [%]	Denoising Bleu [%]
Word			11.2	14.5	17.2
Mikolov et al. [2013c]	threshold	100	11.1	13.7	15.6
		500	11.0	13.7	16.2
		2000	10.7	14.0	16.5
Top frequent	count	50k	12.0	15.7	16.8

Table 6.2: Word embedding vocabulary cut-off

Bleu [%]	20k	50k	100k
50k	11.1	11.3	11.2
100k	11.2	11.2	11.1
150k	10.9	10.9	-

Table 6.3: Phrase embedding vocabulary cut-off

Bleu [%]	50k	100k	150k
50k	11.3	-	-
100k	11.9	11.9	-
150k	12.0	11.9	11.9
200k	12.0	-	-

Chapter 7

Conclusion

For unsupervised machine translation system, the context-aware beam search language model help at the lexicon choice step.
The denoising networks which aimed at the insertion/deletion/reordering noise in the word-by-word translation sentence works well for fertility and localized reordering problem. Even though BPE is known to be an effective way to overcome the rare word problem in standard NMT, BPE embedding performs worse than word embedding in our case especially when vocabulary is small.
Word-by-word translation based on cross-lingual word embedding depends highly on the frequent word mappings. We found that phrase embedding only helps in word-by-word translation with context-aware beam search, it works not as good as that under the processing of denoising autoencoder.

Appendix A

Appendix

List of Figures

List of Tables

Bibliography

Oliver Adams, Adam Makarucha, Graham Neubig, Steven Bird, and Trevor Cohn. Cross-lingual word embeddings for low-resource language modeling. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, volume 1, pages 937–947, 2017.

Mikel Artetxe, Gorka Labaka, and Eneko Agirre. Learning bilingual word embeddings with (almost) no bilingual data. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 451–462, 2017a.

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. Unsupervised neural machine translation. *arXiv preprint arXiv:1710.11041*, 2017b.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.

Hailong Cao, Tiejun Zhao, Shu Zhang, and Yao Meng. A distribution-based model to learn bilingual word embeddings. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 1818–1827, 2016.

Yong Cheng, Wei Xu, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. Semi-supervised learning for neural machine translation. *arXiv preprint arXiv:1606.04596*, 2016.

Trevor Cohn and Mirella Lapata. Machine translation by triangulation: Making effective use of multi-parallel corpora. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 728–735, 2007.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data. *arXiv preprint arXiv:1710.04087*, 2017.

Jocelyn Coulmance, Jean-Marc Marty, Guillaume Wenzek, and Amine Benhalloum. Trans-gram, fast cross-lingual word-embeddings. *arXiv preprint arXiv:1601.02502*, 2016.

Long Duong, Hiroshi Kanayama, Tengfei Ma, Steven Bird, and Trevor Cohn. Learning crosslingual word embeddings without bilingual corpora. *arXiv preprint arXiv:1606.09403*, 2016.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. Convolutional sequence to sequence learning. *arXiv preprint arXiv:1705.03122*, 2017.

Stephan Gouws, Yoshua Bengio, and Greg Corrado. Bilbowa: Fast bilingual distributed representations without word alignments. In *International Conference on Machine Learning*, pages 748–756, 2015.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tieyan Liu, and Wei-Ying Ma. Dual learning for machine translation. In *Advances in Neural Information Processing Systems*, pages 820–828, 2016.

Yedid Hoshen and Lior Wolf. An iterative closest point method for unsupervised word translation. *CoRR*, abs/1801.06126, 2018. URL `http://arxiv.org/abs/1801.06126`.

Yunsu Kim, Julian Schamper, and Hermann Ney. Unsupervised training for large vocabulary translation using sparse lexicon and word classes. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, volume 2, pages 650–656, 2017.

Philipp Koehn. *Statistical machine translation.* Cambridge University Press, 2009.

Guillaume Lample, Ludovic Denoyer, and Marc'Aurelio Ranzato. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043*, 2017.

Guillaume Lample, Myle Ott, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. Phrase-based & neural unsupervised machine translation. *arXiv preprint arXiv:1804.07755*, 2018.

Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015a.

Thang Luong, Hieu Pham, and Christopher D Manning. Bilingual word representations with monolingual quality in mind. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*, pages 151–159, 2015b.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013a.

Tomas Mikolov, Quoc V Le, and Ilya Sutskever. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*, 2013b.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013c.

Malte Nuhn and Hermann Ney. Em decipherment for large vocabularies. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 759–764, 2014.

Malte Nuhn, Arne Mauser, and Hermann Ney. Deciphering foreign language by combining language models and context vectors. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 156–164. Association for Computational Linguistics, 2012.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

Sujith Ravi and Kevin Knight. Deciphering foreign language. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 12–21. Association for Computational Linguistics, 2011.

Sebastian Ruder, Ivan Vulić, and Anders Søgaard. A survey of cross-lingual word embedding models. *arXiv preprint arXiv:1706.04902*, 2017.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

Ivan Vulic and Anna-Leena Korhonen. On the role of seed lexicons in learning bilingual word embeddings. 2016.

Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. Normalized word embedding and orthogonal transform for bilingual word translation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1006–1011, 2015.

Meng Zhang, Yang Liu, Huanbo Luan, and Maosong Sun. Adversarial training for unsupervised bilingual lexicon induction. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1959–1970, 2017.