Rheinisch-Westfälische Technische Hochschule Aachen
Lehrstuhl für Informatik 6
Prof. Dr.-Ing. Hermann Ney

Seminar Selected Topics in Human Language Technology and Pattern Recognition im SS 2016

Attention-based Neural Machine Translation

Derui ZHU

Matrikelnummer 352273

Betreuer: Parnia Bahar

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

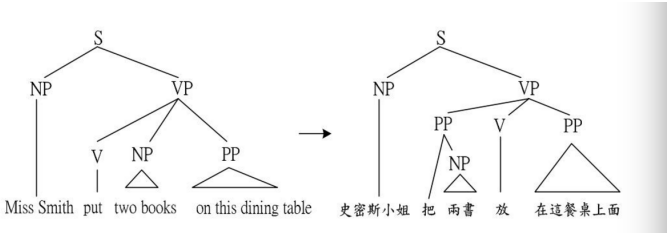

Abbildung 1: Example of rule-based translation process

1 Introduction

Machine translation is one important research field of natural language processing. In general, machine translation is the process of transfering one natural language into another natural language. From early lexicon translation to corpus-based statistical translaton, machine translation technologies experience a huge improvement.
Recalling the history of machine translation development, before 1980s, rule-based translation were the most important approaches. In the end of 1980s, statistical translation approaches were proposed, the situation changed. It rapidly becomes mainstream approaches. Recently, neural network-based translation has shown its huge power.

1.1 Rule-based Translation

Rule-based translation is a process that generate target language by analysing source language's structure and building a meaningful representation. Rule-based approaches are mainly divided into transfer-based and interlingua-based translation process. Interlingua-based approach generate intermediate language first after analysing source language. Then the target language is produced by the intermediate langauge. Most of the rule-based translation system uses transfer-based translation process. It is composed of three steps. Firstly, the system generates an abstract representation of source language. Then, the abstract representation of source language is transferred into the abstraction representation of target language. In the end, the abstraction representation of target language generates target language.
For example, we consider to translate the english sentence "Miss Smith put two books on this dining tableïnto chinese. First of all, the system praser its morphological syntax to get the syntax tree(below [2]). During the transfer phase, lexical and syntax transform are excuted at the same time. After that, we get the target words, which locate in the nodes of syntax tree generated.

1.2 Stastical-based Translation

The basic idea of statistical approach is to obtain translation rules and probability parameters automatically from bilingual corpus by employing statistical machine learning technologies. Graphic 1 shows string-based and tree-based translation model, which are used in statistical translation widely. Statistical translation models usually is partitioned as word-based, phrase-based and gramma-based translation model. Phrase-based translation model is a mature tranlation approach. The phrase means successive word sequences rather than the phrase defined in linguistics. The main idea of this model is to extract

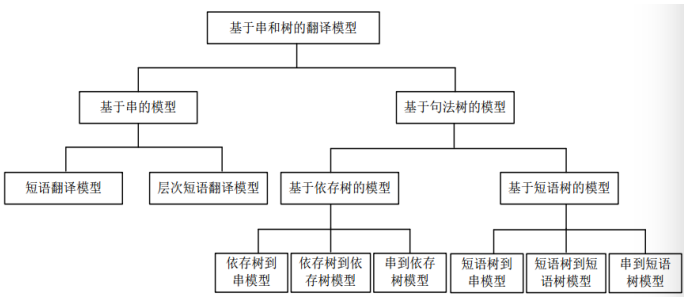

Abbildung 2: string-based and tree-based translation models

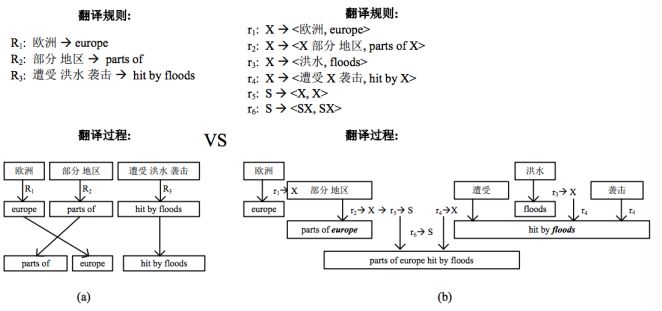

Abbildung 3: The compare between phrase-based model(a) and hierarchical phrase-based model

phrase-to-phrase translation rules from large number of bilingual sentences. Then, in the phase of tranlastion, system cuts the words to translate according to the phrases that got by last step. By the help of re-order model, finally, target sentence is generated. If there is variables in the phrase contained into translation rules, the phrase-based model is hierarchical phrase-based translation model. It owns better representation ability and performance. The below graph 3 show the comparative between these two models.

1.3 Neural Network-based Translation

Neural network-based translation approach is proposed recently. Based on the sequence to sequence learning, neural machine translation doesnot rely on the specialized knowledge on langauge, like gramma, syntax, sentence structure and so on. Traditional statistical machine translation has much dependency on sub model, for example, rescoring model and reorder model. Compared to that, neural machine translation is much easier. However, the complexity of training neural translation model is harder than statistical translation model.

As shown in the graph 4, neural translation process is divided into two steps, encoder and decoder. In the phase of encode, it encodes the source sentence into a vector that contains source sentence information. Then, the decoder generate the target sentence based on the vector.

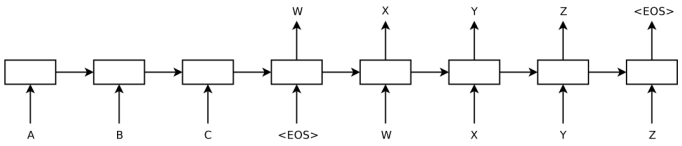

Abbildung 4: Sequence to sequence Tranlation

1.4 Matrics on translation performance

Not only is it not easy to translate natural language, but also it is really hard on evaluating translating quality. Most of paper use Bleu as metric on translation quality, which proposed by IBM.
Bleu(Bilingual Evaluation understudy) is used for measuring the quality of translation by comparing the similarity between machine's result and that of a human. The closer a sentence translated by machine is to human's translation, the better it is. How to compute a Bleu score? Firstly, we count the maximun possible numbers of a n-gram word that display in sentence translated by machine and professional result. Then, we take the minimun value from last two numbers. After that, we divide the value by the number of n-gram word in the professional result. The formula is shown below.

$$Count_{clip}(n-gram) = min\{Count(n-gram), \quad MaxRefCount(n-gram)\}$$

$$p_n = \frac{\sum_{C \in candidates} \sum_{n-gram \in C} Count_{clip}(n-gram)}{\sum_{C \in candidates} \sum_{n-gram \in C} Count(n-gram)}$$

$$Bleu = BP * exp[(\sum_{n=1}^{N} w_n log p_n)]$$

$$BP = \begin{cases} 1 & \text{if } c > r \\ EXP(1-r/c)) & \text{if } c <= r \end{cases}$$

This paper will be organized as 4 parts in the rest sections. The second part will give a brief introduction on neural network, which can be used in sequence-to-sequence learning. It is the foundation of neural machine translaton. Next, the third part will propose a classical neural machine translation model. Then, the attention-based neural machine translation will be shown. In the last part, we will show the experiment result between these machine translation model in terms of Bleu scores.

2 Neural Nerwork

Deep Neural Network is an extremely powerful approach for machine translation, image and speech recognition and so on. Along with the development of computing resources and related technologies, large scale deep neural network training is becoming more and more realistic. In principle, if given enough data and computing resource, neural network can fit any real-world model.

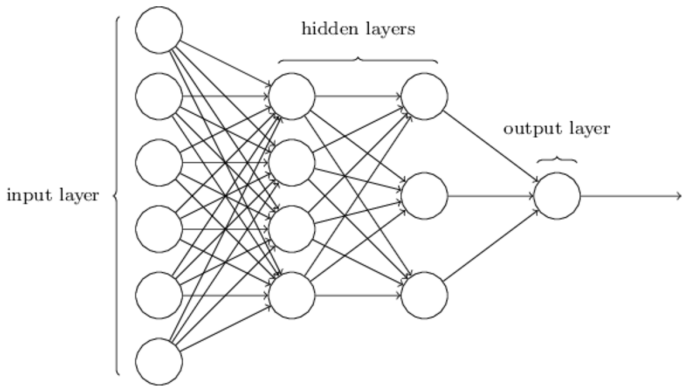

Abbildung 5: Basic Structure of Deep Neural Network

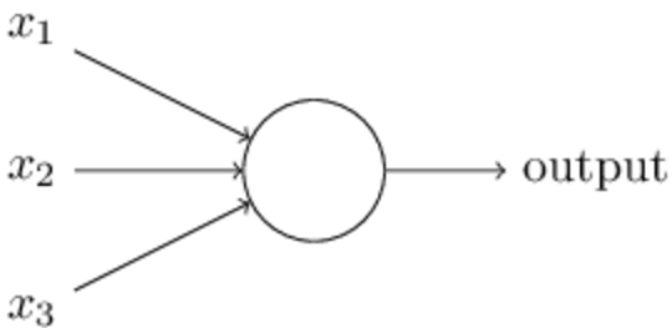

Abbildung 6: Perceptron

2.1 Brief Introduction

In general, neural network is consisted of input layer, hidden layer and output layer(see below graph). The hidden layer usually has multiple layers. Each layer(except input layer) is composed of multiple neurons called perceptrons. Perceptrons were proposed by Frank Rosenblatt [], which inspired by the work of Warren McCulloch and Walter Pitts[]. It usually means sigmoid function. How does it work? One perceptrons(see below graph) accepts some scalar input to produce a score by linear tranformation, then output a value between 0 and 1 by non-linear transformation. The linear Transformation follows the rule below, where W is the weight matrix and b is their biases. The non-linear transformation use the sigmoid function(see below). Overall, a general neural network can be decribed as below, which h is a non-linear transformation function. Different non-linear function may affect the model performance seriously.

$$S = W^T + b$$

$$y = \frac{1}{1 + e^{-S}}$$

$$y = h(W^T + b)$$

2.2 Backpropagation Algorithm

Any feed-forward neural network can be trained with backpropagation algorithm, as long as the cost function are differentiable. The most frequently used cost function is the summed squared error(SSE), defined as:

$$C = \frac{1}{1}\sum_{p}^{n}\sum_{k}^{o}(d_{pk} - y_{pk})$$

where d is the desired output, n is the total number of training samples and o is the number of output units.
Backpropagation algorithm is actually a propagated gradient descent algorithm, where gradients are propagated backward, leading to very efficient computing of the higher layer weight change. According to the gradient descent, each weight change in the network should be proportional to the negative gradient of the cost function, with respect to the specific weight. The cost function that applid SSE should be defined as:

$$e_n = (y_n - NNet(X_n))^2 = (y_n - s_1^{(L)})^2 = (y_n - \sum_{i=0}^{d^{(L-1)}} w_{i1}^{(L)} x_i^{(L-1)})^2$$

To compute the optimised weight w, we firstly consider the weights associated with output layer. It follows the formula shown below.

$$\begin{aligned}\frac{\partial e_n}{\partial w_{i1}^{(L)}} &= \frac{\partial e_n}{\partial s_1^{(L)}} * \frac{\partial s_1^{(L)}}{\partial w_{i1}^{(L)}} \\ &= -2(y_n - s_l^{(L)}) * (x_i^{(L-1)})\end{aligned}$$

where L is the output layer. If we diffuse to any layer, then, the computing formula will become:

$$\frac{\partial e_n}{\partial w_{ij}^{(l)}} = \frac{\partial e_n}{\partial s_j^{(l)}} * \frac{\partial s_j^{(l)}}{\partial w_{ij}^{(l)}}$$

We set:

$$\delta_j^l = \frac{\partial e_n}{\partial s_1^{(L)}}$$

Then:

$$\frac{\partial e_n}{\partial w_{ij}^{(l)}} = \delta_j^{(l)} * (x_i^{l-1})$$

Therefore, our task is to compute the value of $\delta_j^{(l)}$. It can be inferred as:

$$\begin{aligned}\delta_j^l &= \frac{\partial e_n}{\partial s_j^{(l)}} \\ &= \sum_{k=1}^{d^{(l+1)}} \frac{\partial e_n}{\partial s_k^{(l+1)}} * \frac{\partial s_k^{l+1}}{\partial x_j^{(l)}} * \frac{\partial x_j^{(l)}}{\partial s_j^{(l)}} \\ &= \sum_{k}(\delta_k^{(l+1)})(w_{jk}^{(l+1)})(tanh^{'}(s_j^{(l)}))\end{aligned}$$

$\delta_j^{(l)}$ can be computed backwards from $\delta_{(l+1)}^k$.

2.3 Recurrent Neural Network

Recurrent neural network is a special neural network, which each the wight of each layer are same. Since one important property of translation is the source and target sentences alway are varibale-lenghth. To deal with this problem, recurrent neural network is proposed. Considering a variable-lenth input $\mathbf{X} = (x_1, x_2, ..., x_T)$ abd out output $\mathbf{Y} = (y_1, y_2, ..., y_{T'})$, x_i and y_i is represented by its integer index in a vocabulary. A vocabulary includes the most often n(self-defined) works. Every word in a vocabulary has a distinct number, which arranged by its frequence in the corpus.

3 Neural Machine Translation

From the view of neural network, machine translation can be modeled by a deep recurrent neural network. The translation problem is equal to find a maximun probability target sentence **e** in terms of given source sentence **f**, for example $max_y p(y|x)$. In neural machine translation, we fit a model with weights and biases by applying recurrent network based on bilingual language data set. We usually use a encoder to encode the source sentence into a vector, which represent the source sentence. Then, a decoder decode the vector into a target sentence. The encoder and decoder always are recurrent neural network. Next, we the recurrent

3.1 Recurrent Neural Network

4 Attention-based Neural Machine Translation

5 Evaluation

6 Introduction

Dies ist eine Referenz auf Abschnitt 6

Erzeugen wir erst etwas Text, um uns mit der Schreibweise in LaTeX vertraut zu machen. LaTeX bricht selbständig Zeilen und Seiten um. Selbst in einem Neogloismus wie dem ziemlich langen Wort aus Michael Endes zuletzt geschriebenem Buch „Der satanarchäolügenialkohöllische Wunschpunsch“ findet LaTeX eine gute Trennstelle. Manchmal ist es aber sinnvoll, selber eine Trennstelle vorzugeben. Dies gibt man mit \− im Wort im TeX-File selbst an, also z.B. Trenn\−stelle.

Lässt man eine Zeile im TeX-File frei so wird automatisch die nächste Zeile eingerückt. Wem das nicht gefällt, der kann im TeX-File den Befehl `\parindent0pt` auskommentieren und das damit unterdrücken. Text mit ä, ö, Ü, Straße, Café.

Dies ist ein laufender-Text mit Einsteins Formel $E = m \cdot c^2$.

Tabelle 1: Beispiel Tabelle

links	zentriert	rechts
links2	zentriert2	rechts2
XXXXXXXXXXX	XXXXXXXXXXXXXXXXXXXXX	XXXXXXXXXXXX

Eine etwas kompliziertere Formel:

$$a\frac{1-q^{n+1}}{1-q} = \sum_{i=0}^{n} aq^n \qquad \text{mit} \quad a, q \in \mathcal{R}, q \neq 1$$

Nun folgt die Bayessche Entscheidungsregel:

$$r(X) = \underset{w_1 \ldots w_N}{\operatorname{argmax}} \left\{ Pr(w_1^N | x_1^T) \right\} \quad \text{mit} \quad Pr(w_1^N | x_1^T) = \frac{Pr(x_1^T | w_1^N) \cdot Pr(w_1^N)}{Pr(x_1^T)} \quad . \tag{1}$$

Beispiel für ein lineares Gleichungssystem:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 &= y_1 \\ a_{21}x_1 + a_{22}x_2 &= y_2 \\ a_{31}x_1 &= y_3 \end{aligned}$$

7 Neural Network

7.1 Unterabschnitt

8 Neural Machine Translation

9 Attention-based Neural Machine Translation

9.0.1 Unter-Unterabschnitt: Beispiel für Tabellen

In der Umgebung zum Plazieren von Tabellen (und auch Grafiken) kann man optional die Werte t, b, h und H verwenden. Damit lassen sich Tabellen am oberen Rand (top), unten (bottom) und an der aktuellen Position (here) plazieren. Je nach Seitenlayout funktioniert die Angabe h nicht immer. Mittels H lässt sich dann die aktuelle Position erzwingen.

9.1 Unterabschnitt: Beispiel für Grafiken

Grafiken werden üblicherweise mit dem (frei verfügbaren) Programm tgif erstellt. Alternativ hierzu lässt sich auch das Programm xfig verwenden. tgif ist zwar nicht ganz einfach zu bedienen, stellt dem (erfahrenen) Anwender aber wesentlich mehr Möglichkeiten zur

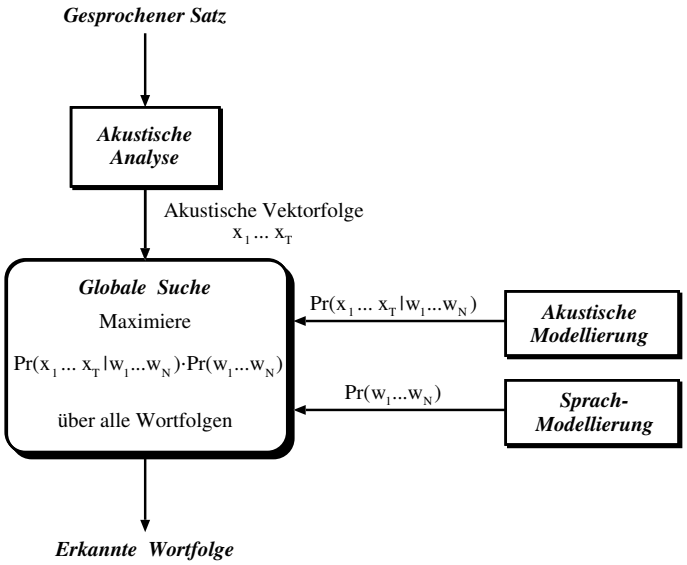

Abbildung 7: Architektur eines Spracherkennungssystems

Gestaltung von Grafiken zur Verügung. Die Konvertierung der *.obj Dateien nach encapsulated PostScript erfolgt über die Kommandozeile mit

tgif -print -eps <Datei>.obj

Nun erzeugen wir mittels BibTeX eine Literaturliste. Dazu erzeugen wir uns ein File namens <Dateiname>.bib, dass denselben Dateinamen wie unser TeX-File hat. Dieses wird dann mit

bibtex <Dateiname>

compiliert. Die im laufenden Text vorkommenden symbolischen Literaturverweise zeigen dann auf den entsprechenden Eintrag im Literaturverzeichnis. Dies geschieht mit dem Befehl \cite{}. Das sieht dann z.B. wie folgt aus.

Zitat aus [1], Zitat aus [3], Zitat aus [2]. Wichtig ist, das zuerst das TeX-File compiliert wird und dann das BibTeX-File. Da sich dann die Referenzen meist ändern, ist ein erneutes Compilieren des TeX-Files notwendig.

Es gibt zahlreiche style-files, mit denen man das Layout eines Literaturverzeichnisses ändern und die Darstellung der Literatur-Refererenzen modifizieren kann.

Letzter Hinweis: LaTeX-Neulinge sollten auch die Kommentare im TeX-File lesen.

10 Evaluation

Literatur

[1] Autor1, Autor2, and Autor3. Titel des Papers. In *Name des Bandes, in dem Artikel veröffentlicht wurde*, volume Band, pages xxx–yyy, Monat 1991.

[2] Autor1, Autor2, and Autor3. Titel des Reports. Technical report, Name des Instituts, Monat 1996.

[3] M. Ende. *Der Wunschpunsch.* Tiemann Verlag, 1989.