

MT Praktikum - Alignment

6. Juni 2017

Um die Aufgaben auszuführen, können Sie Ihre Daten in folgendem Verzeichnis speichern:
`/project/smtstud/ss17/systems/USERNAME/`

Wir werden zunächst mit Hilfe des GIZA++ Toolkits: IBM Model 1, HMM, IBM Model 3 und IBM Model 4 trainieren und uns das resultierende Word Alignment ansehen. Dann werden wir das Training für ein IBM Model 1 Lexikon implementieren und Trainingsiterationen an einem konstruierten Beispiel durchgehen.

Sehen Sie sich die Trainingsdaten in deutsch und englisch einmal an, um einen Eindruck davon zu gewinnen, womit Sie arbeiten:

`/project/smtstud/ss17/data/corpus/train.de-en.1-99.de`
`/project/smtstud/ss17/data/corpus/train.de-en.1-99.en`

Bitte melden Sie sich nach Abschluss jeder Aufgabe (oder ggf. Zwischenaufgabe), sodass wir ausschließen können, dass Sie mit fehlerhaften Dateien weitermachen. Melden Sie sich bitte auch, wenn eine angegebene Laufzeit für ein Programm stark überschritten wird.

1. Trainieren der Word Alignment Models: Vorbereitung der Daten

Die parallelen Trainingsdaten müssen zunächst vorbereitet werden, da das GIZA++ Toolkit sie in einem bestimmten Format erwartet.

(a) Führen Sie das script:

`/project/smtstud/ss17/bin/prepare-data.sh`

in Ihrem Verzeichnis aus und sehen Sie sich die Ausgabe und die erzeugten Dateien an. (Laufzeit ca. 4min, hauptsächlich in Schritt 1.1)

- Schritt 1.0.5 bezieht sich auf das Trainieren faktorisierter Modelle, welche wir heute nicht verwenden werden. (D.h. dieser Schritt hat auf unsere nicht faktorierten Daten keine Wirkung. Faktorisierte Übersetzungsmodelle werden in der Vorlesung später besprochen.)

- Schritt 1.1 berechnet die Wortklassen, die für das Trainieren von IBM Model 4 benötigt werden. Auf diesen Schritt werden wir heute nicht weiter eingehen.

- Die in Schritt 2.1a erstellten cooccurrence files listen alle de-en Wortpaare auf, die in den Trainingsdaten zusammen auftauchen. Die Dateien sind die Vorlage für die Lexika.

Sehen Sie sich die Dateien aus Schritt 1.2 und 1.3 (`./corpus/de.vcb`, `./corpus/en.vcb`, `./corpus/de-en-int-train.snt`) einmal an.

- Was tun die Schritte 1.2 und 1.3?
- Wieso wird der Trainingscorpus so bearbeitet?

2. Trainieren der Word Alignment Modelle bis zu IBM Model 4

Verwenden Sie zum Trainieren der word alignment models das Programm:

```
/project/smtstud/ss17/bin/giza/GIZA++
```

(Sie können sich alle möglichen Parameter mit ihren default-Werten ansehen, wenn Sie das Programm ohne Parameter aufrufen. Wir werden nicht die Zeit haben diese alle zu besprechen.)

Berechnen Sie die Modelle zunächst in der Sprachrichtung: Quellsprache: Englisch, Zielsprache: Deutsch.

Hinweis: aus historischen Gründen (noisy channel model) werden sich die Ausgabe-dateien hierfür im Verzeichnis: „giza.de-en“ befinden und auch das Dateipräfix de-en haben.

Verwenden Sie folgende Parameter:

(Hinweis: damit Sie nicht all diese Parameter abtippen müssen, können Sie sich den Programmaufruf hier kopieren: `/project/smtstud/ss17/bin/run.giza` – Führen Sie den Befehl in Ihrem Userverzeichnis aus, wo Sie die Daten aus Aufgabe 1 vorbereitet haben. Laufzeit ca. 6 min)

Vorbereitete Trainingsdaten einlesen:

```
-CooccurrenceFile ./giza.de-en/de-en.cooc  
-c ./corpus/de-en-int-train.snt  
-s ./corpus/en.vcb  
-t ./corpus/de.vcb
```

Anzahl der Iterationen für jedes Modell:

```
-hmmiterations 5
```

```
-model1iterations 5
-model2iterations 0 (wir verwenden das HMM model anstelle von model 2)
-model3iterations 3
-model4iterations 3
```

Alle Zwischenmodelle ausgeben (wir werden uns einige davon ansehen)

```
-model1dumpfrequency 1
-hmmdumpfrequency 1
-model2dumpfrequency 1
-model345dumpfrequency 1
-transferdumpfrequency 1
```

Ausgabeprefix:

```
-o ./giza.de-en/de-en
```

Bedeutung der Dateinamen der Ausgabedateien:

Datei: ./giza.t-s/t-s.xn.m

t: target language (e.g. de)

s: source language (e.g. en)

x: type of table: t = word lexicon, a = alignment model, A = alignment, n = fertilities, d = distortion model, p0 = probability, that the NULL word is not inserted

n: model number (z.B. model 1 oder hmm, aus historischen Implementationsgründen sind model 4 files zum Teil als höhere model 3 Iterationen geführt, z.B. de-en.A3.5 ist tatsächlich das Alignment der 2. Iteration model 4)

m: iteration number

- (a) Sehen Sie sich die Perplexität der Trainingsdaten für die verschiedenen Modelle und Iterationen an. (Hinweis: diese finden Sie sowohl in der Logdatei des Trainings train.giza.log als auch zusammengefasst in der Datei giza.de-en/de-en.perp)

- Wie entwickelt sich die Perplexität der Trainingsdaten und warum?

- (b) Sehen Sie sich das Wortlexikon aus dem Model 1 Training Iteration 5 (giza.de-en/de-en.t1.5) und aus dem Model 4 Training (giza.de-en/de-en.t3.final) an. (Lexikonformat: EN DE probability)

- Finden Sie die jeweils die 3 wahrscheinlichsten Übersetzungen für das englische „you“.

Hinweis:

1. Verwenden Sie die beiden Vokabularien: giza.de-en/de-en.trn.src.vcb und giza.de-en/de-en.trn.trg.vcb.

2. Befehl: `grep '^<Index>' <file> |LC_ALL=C sort -g -k <key>`

model 1		model 4	

- Wie unterscheiden sich die Lexika und warum?
- (c) Vergleichen Sie die Alignment Dateien aus der jeweils letzten Iteration von Model 1 Training (`giza.de-en/de-en.A1.5`) und aus dem Model 4 Training (`giza.de-en/de-en.A3.final`).
- Finden Sie das Alignment für Satz Nummer 11 in beiden Dateien und tragen Sie die alignment links jeweils unten ein:

Model 1 Training (Iteration 5) `giza.de-en/de-en.A1.5`

0 1 2 3 4 5 6 7 8 9 10 11 12
 NULL “ Noh ” is a typical , traditional form of theatre .

“ Noh ” ist eine typische , traditionelle Form des Theaters .
 0 1 2 3 4 5 6 7 8 9 10 11 12

Model 4 Training (Iteration 3) `giza.de-en/de-en.A3.final`

0 1 2 3 4 5 6 7 8 9 10 11 12
 NULL “ Noh ” is a typical , traditional form of theatre .

“ Noh ” ist eine typische , traditionelle Form des Theaters .
 0 1 2 3 4 5 6 7 8 9 10 11 12

- Was können Sie hier ablesen?

Trainieren Sie nun alle Modelle für die umgekehrte Sprachrichtung: Führen Sie den Befehl in `/project/smtstud/ss17/bin/run.giza.rev` in Ihrem User-verzeichnis aus, wo Sie die Daten aus Aufgabe 1 vorbereitet haben. (Laufzeit ca. 6 min)

- (d) Vergleichen Sie die Alignment Dateien aus der letzten Iteration model 1 aus den beiden Trainingsrichtungen. (`giza.de-en/de-en.A1.5` und `giza.en-de/en-de.A1.5`)

- Finden Sie das Alignment für Satz Nummer 106855 (am Ende der Dateien) in beiden Dateien und tragen Sie die alignment links jeweils unten ein:

Model 1 Training (Iteration 5) forward giza.de-en/de-en.A1.5

0	1	2	3	4	5	6
NULL	may	I	take	a	message	?

	soll	ich	eine	Nachricht	aufschreiben	?
0	1	2	3	4	5	6

Model 1 Training (Iteration 5) reverse giza.en-de/en-de.A1.5

0	1	2	3	4	5	6
NULL	may	I	take	a	message	?

	soll	ich	eine	Nachricht	aufschreiben	?
0	1	2	3	4	5	6

- Was lässt sich an diesem Beispiel deutlich ablesen?

3. Implementierung: IBM model 1 (Zusatzaufgabe)

Das IBM1 Modell wird benutzt um ein word alignment und ein Lexikon für einen Satz-alinierten Corpus zu generieren. Schreiben sie ein Programm, das ein Lexikon und einen Corpus als Eingabe bekommt und danach einen Schritt des EM-Algorithmus zum Trainieren des IBM1 Modell vollführt.

Falls Sie das in Perl/Python programmieren möchten, können Sie folgende Vorlage verwenden: `/project/smtstud/ss17/bin/IBM1.pl` oder `/project/smtstud/ss17/bin/IBM1.py` (Hinweis: Nehmen Sie den Pseudocode aus der Vorlesung zur Hilfe.)

Das initiale Lexikon und die Beispielsätze aus der Vorlesung finden sie im Verzeichnis: `/project/smtstud/ss17/data/smallExample`

- Wie verändern sich die lexikalischen Wahrscheinlichkeiten mit den Iterationen?
- Wie viele Iterationen scheinen Ihnen in diesem Fall sinnvoll?

DE	EN	init	iteration 1	iteration 2	iteration 3	...	iteration?
Haus	house						
Haus	the						
das	house						
das	the						
das	book						
ein	a						
ein	book						
Buch	the						
Buch	a						
Buch	book						