Human Language Technology:
Applications to Information Access

Lesson 6a: Appendix on Text Alignment at the Sentence and Word Levels

November 3, 2016

EPFL Doctoral Course EE-724
Andrei Popescu-Belis
Idiap Research Institute, Martigny

Plan of the lesson

- Learning a translation model requires pairs of sentences which are translations of each other
  – to be obtained from documents and their translation by sentence alignment
- Other tasks require automatic word-level alignment in pairs of parallel sentences
  – can be derived from the process of building translation models, e.g. with IBM Models

Text alignment (1/2)

- Problem
  - start with sets of English and French documents which are translations of one another done by human translators
  - identify pairs of translated documents
    → document alignment
  - in such documents, identify pairs of translated sentences
    → sentence alignment
  - in such sentences, identify links between translated words
    → word alignment

- Why is it difficult?
  - documents: naming conventions might differ
  - sentences: translators do not always translate sentence by sentence (especially long ones), and might omit sentences
  - words: translation is not word-by-word

Text alignment (2/2)

- Expected quality of automatic alignment depends on human translation quality
- Important notion for sentence alignment: ‘bead’
  - group of one or more English sentences, which are exact translations of a French group
  - on EN/FR, about 90% of alignments are 1:1
    - not obvious to find because alignment mistakes propagate
  - also 1:2 or 2:1, even 3:1 or 1:3, 2:2 (mixed sentences), 1:0 and 0:1 (omissions), etc.

Sentence alignment using length in characters (Gale and Church 1993)

- Find most likely alignment of ***TE*** and ***TF*** texts
  - sentences: $\boldsymbol{TE}_{1..E} = (e_1, \ldots, e_i, \ldots, e_E)$ and $\boldsymbol{TF}_{1..F} = (f_1, \ldots f_j, \ldots, f_F)$
  - notations often used in SMT papers, *F* for French or foreign
- Note $D(i, j)$ the lowest cost of alignment of $\boldsymbol{TE}_{1..i}$ and $\boldsymbol{TF}_{1..j}$
  - alignment is a set of tuples or beads $\{((e_i, \ldots), (f_j, \ldots)), \ldots\}$
  - for instance $((e_i), (f_j))$ is a 1:1 bead, $((e_i, e_{i+1}), (f_j))$ is a 2:1 bead
    - Gale and Church only consider {1:1, 0:1, 1:0, 2:1, 1:2, 2:2}
- If we can compute $D(E, F)$ recursively, then we find:
  - the lowest costs for all $D(i, j)$, $1 \leq i \leq E$ and $1 \leq j \leq F$
  - the best alignment by looking at how $D(E, F)$ was computed

Recursive definition of *D*

$$D(i, j) = \min \begin{cases} D(i, j\text{-}1) \quad + cost_{0:1}(\emptyset, f_j) \\ D(i\text{-}1, j) \quad + cost_{1:0}(e_i, \emptyset) \\ D(i\text{-}1, j\text{-}1) + cost_{1:1}(e_i, f_j) \\ D(i\text{-}1, j\text{-}2) + cost_{1:2}(e_i, (f_{j\,-1}, f_j)) \\ D(i\text{-}2, j\text{-}1) + cost_{2:1}((e_{i\text{-}1}, e_i), f_j) \\ D(i\text{-}2, j\text{-}2) + cost_{2:2}((e_{i\text{-}1}, e_i), (f_{j\,-1}, f_j)) \end{cases}$$

- Implemented using dynamic programming
  - quadratic complexity, but run only on paragraphs, so OK
- How do we estimate each $cost_{X:Y}((\ldots), (\ldots))$?
  - look at training data and consider the observed ratio of characters between aligned sentences for each X:Y

Estimating the cost of alignments

- Idea: the cost for each bead type X:Y is related to the probability of the type given the two lengths of the sentences in the bead
- Let *LE* be the length in characters of the English side of a bead (e_i, …) and *LF* the length of the French side of it (f_j, …)
  - how does the difference *LE*-*LF* compare to the average distance of correct beads?
    - e.g., French sentences have on average more words than English ones
  - average ratio μ and STD (s^2) can be estimated on aligned data
  - comparison of *LE* and *LF* using $d = (LF - \mu LE)/\mathrm{sqrt}(LF.s^2)$
- Therefore $cost_{X:Y}((\ldots), (\ldots)) = cost_{X:Y}(LE, LF) = -\log P(\mathrm{X{:}Y} \mid d) =$
  $= -\log P(\mathrm{X{:}Y}) - \log P(d \mid \mathrm{X{:}Y}) + \log P(d)$
  - these probabilities can be estimated from training data

Results of Gale and Church (1993)

- On pairs with English, French and German
- About 4% error rate for the described method
  - over 1:1 alignments, only 2% error rate
- Method to compute alignment confidence
  - selected 80% of corpus with only 0.7% error rate

Sentence alignment using the lexicon

- Lexical matching
  - identify translational equivalents = anchor point candidates
  - optimize alignments between sentences based on anchors

- Kay and Röscheisen (1993)
  - start with initial anchors: first and last sentences
  - iterate
    - form an envelope of possible alignments given anchors
    - find pairs of words that occur in the partial alignments
    - find pairs of sentences which contain many such words and add them to the set of anchors
- Performance
    - after iterations, 96% correct – but computationally intensive

Word alignment

- In a pair of sentences (e, f) which are translations of each other, find which word in e is translated into which word in f, and vice-versa
  - alignment point = pair of words which are translations of each other

- Defining the correct alignment is difficult even for humans
  - one-to-many, many-to-one, idioms, words with no equivalents
  - best option: define **sure** alignment points (S) and **possible** alignment points (P), with $S \subseteq P$

- Measuring the quality of an alignment A: alignment error rate
  - **recall**=$|A \cap S|/|S|$ and **precision**=$|A \cap P|/|A|$ and
    $\mathbf{AER}_{S,P}(A) = 1 - (|A \cap S| + |A \cap P|) / (|A|+|S|)$
  - AER=0 if A gets all sure points and zero or more possible points

Using IBM Models for word alignment

- Results of IBM Models after EM algorithm = probabilities for lexical translation and alignment

- Can be used to determine the most probable word alignment for each sentence pair ("Viterbi alignment")
  - Model 1, for each word e_i select the word f_j that has maximal probability $t(e_i|f_j)$
  - Model 2, same but maximize $t(e_i|f_j)\ P_a(j|i, E, F)$
  - Models 3-5, no closed form expression
    - start with Model 2, then use some heuristics to improve it

Why is it called "Viterbi alignment"?

- Viterbi algorithm (VA)
  - proposed by Andrew Viterbi for decoding in signal processing
  - find most likely sequence of hidden states that explain an observation
    - this is called the Viterbi path
    - especially for Hidden Markov Models (HMMs)

- Automatic speech recognition
  - VA is used to find the most likely (forced) alignment between audio and words, using HMMs previously trained on transcribed audio

- Natural language processing
  - Viterbi alignment = most likely alignment, even if not found using VA
  - for word alignment in MT
    - IBM Models: no HMMs, but the most likely alignment is still called "Viterbi"
    - HMM models: can use VA or other dynamic programming techniques

Improving word alignments

- For a given translation direction, this approach can find one-to-one alignments, multiple-to-one, one-to-zero, but *never one-to-multiple*
  - still, for a correct alignment, we might need both
  - {*Paul*} {*was waiting*} {*inside*} $\leftrightarrow$ {*Paul*} {*attendait*} {*à l ' intérieur*}
- Solution: symmetrization, by running algorithm in both directions
  - consider the intersection of the two sets of alignment points, or their union, or enrich intersection with some points of the union
- Many other methods exist for *word alignment*
  - generative: train HMMs on linking probabilities, then use Viterbi decoding or another dynamic programming method
  - discriminative: structured prediction, feature functions, etc.
  - still, for phrase-based translation, IBM Models 1-4 perform well

Applications of word alignment

- Building bilingual dictionaries
- Extracting lexical semantics
- Multilingual word sense disambiguation
- Computer-assisted language learning
- Learning translation models
  - IBM models (no longer state-of-the-art)
    - powerful alignment tool: GIZA++ (Och and Ney 2000)
  - phrase-based translation models

References

- Jörg Tiedemann, *Bitext Alignment*, Morgan & Claypool, 2011
  - available online via EPFL library server
- William Gale and Kenneth Church, “A program for aligning sentences in bilingual corpora”, *Computational Linguistics*, 19(1), p.75-102, 1993
  - classic method for length-based sentence alignment
- Martin Kay and Martin Röscheisen, “Text-Translation Alignment”, *Computational Linguistics*, 19(1), p. 121-142, 1993
  - classic method for lexical sentence alignment
- Philipp Koehn, *Statistical Machine Translation*, Cambridge University Press, 2010, chapter 4 (especially 4.5)