DECODING COMPLEXITY IN WORD REPLACEMENT TRANSLATION MODELS

Kevin Knight, 1999
University of Southern California

Presented by
Eeshan Malhotra
Ankith MS

INTRODUCTION

Statistical models are by far the most prevalent approach for machine translation today (sometimes augmented with other techniques)

We have excellent strategies for *learning* from the corpus. But given a new sentence in target language, the best known decoding algorithms are exponential.
Can better algorithms exist?

INTRODUCTION

We know a much better algorithm for POS tag decoding (Which one?).

What aspect of MT makes this algorithm intractable?

INTRODUCTION

We know a much better algorithm for POS tag decoding (Which one?).

What aspect of MT makes this algorithm intractable?

Word Alignment. If alignment were not a problem, a Viterbi implementation could solve SMT decoding extremely quickly

This paper considers a simple *word-replacement* model, and tries to find if there are fast algorithms for decoding. Or if decoding is doomed to be intractable.

WORD REPLACEMENT MODEL

The basic source-channel framework presumes this approach for generation:

1. English sentences produced by a stochastic process
2. The sentence is translated to French by using another stochastic process

Additionally, for a word replacement model, the stochastic process in step two involves replacement of English words by one or more French words

DECODING

Decoding is the step of generating an English sentence from a new, unseen French sentence. More specifically, in our case, we want the English sentence that was *most likely to have generated* the French sentence

i.e.

$\text{argmax } P(e|f)$

$= \text{argmax } P(e).P(f|e)$

As we have seen many times, P(e) is the language (or source) model, and P(f|e) is the translation (or channel) model

LANGUAGE MODEL

We use the bigram assumption for simplifying the language model.

Thus,

$$P(e) = \prod_{i=1}^{l} P(e_i \,|e_{i-1})$$

This is learn in the bigram model $\mathrm{b}(e_i \,|e_{i-1})$

TRANSLATION MODEL

$$P(f|e) = P(f|e,l)$$
$$= \sum_a P(f,a|e,l)$$

$$P(f,a\,|e,l) = \sum_m P(f,a\,,m|\,e,l)$$
$$= P(f,a,m\,|e,l)$$
$$= P(m|e,l)\,.\,P(f,a\,|e,l,m)$$

Using the derivation done in class, the second term reduces to

$$\prod_{i=1}^{m} [P(a_i|f_1^{i-1}\,, a_1^{i-1}, e, l, m)\,.\,P(f_i\left|f_1^{i-1}, a_1^{i}, e, l, m\right)]$$

TRANSLATION MODEL

Now Model 1 makes some simplifying assumptions.

1. m depends only on l

   $P(m|e,l) = P(m|\ l)$

   This is learnt in a table ∈(m|l)

2. All alignments are equally likely

$$P(a|\ \dots) = {}^{1}/_{l^m}$$

3. $f_i$depends only on e_{a_i}

   $P(f_i\,|f_1^{i-1}, a_1^i, e, l, m) = P(f_i\,|e_{a_i})$

   This is learnt in a table s(f|e) using an EM learning approach

So our parameters are: $\mathrm{b}(e_i\,|e_{i-1})$, ∈(m|l), s(f|e). It is clear that these tables can be built from the training corpus. So let's move to the decoding part.

NAÏVE DECODING

Naïve way to find translations:

1. Enumerate all possible alignments and all possible word replacements.
2. Now pick the one with highest $P(e).P(f|e)$

Number of alignments: l^m

Number of word replacements: (potentially) v^m

(where v is the total English vocabulary size)

This is exponential. But as we will see soon, The best algorithm is also (probably) exponential.

PROVING HARDNESS

NP Complete

A problem that is NP-Hard, and in NP, is NP-Complete

NP Hard

NP-Hard includes some of the hardest known problems. No polynomial time algorithm is known for these problems. It is widely believed that no such algorithm exists

Why do we want to prove NP hardness of decoding?

PROVING HARDNESS

Concepts like NP and NP complete are defined only for decision problems. Consider these two problems:

PROBLEM 1: DECODING (optimization)

Given a string f of length m and the parameter tables $(b, \in, s)$, return a string e of length $l < 2m$ that maximizes $P(e/f)$, or equivalently maximizes

$$
\begin{aligned}
\mathrm{P}(\mathrm{e}) \cdot \mathrm{P}(\mathrm{f}|\mathrm{e}) \quad &= \quad \mathrm{b}(e_1 \mid \text{boundary}) \cdot \mathrm{b}(\text{boundary} \mid e_l) \cdot \prod_{i=2}^{l} \mathrm{b}(e_i|e_{i-1}) \\
&\cdot\ \epsilon(m|l)\ \frac{1}{l^m} \prod_{j=1}^{m} \sum_{i=1}^{l} \mathrm{s}(f_j|e_i)
\end{aligned}
$$

PROVING HARDNESS

Concepts like NP and NP complete are defined only for decision problems. Consider these two problems:

PROBLEM 2: DECODING (decision)

Given a string f of length m and the parameter tables $(b, \in, s)$, and a real number k, does there exist a string e of length $l < 2m$ such that $P(e) \cdot P(f|e) > k$?

These two problems are intimately linked

It is clear that the optimization problem is *at least as hard* as the decision problem.

So, if we prove hardness of the decision version, we can prove the optimization problem to be hard.

DECODING IS IN NP

A problem is in NP if, given an input and a solution, the solution can be verified in polynomial time

Let's work with the decision problem.

It is easy to see that decoding (decision) is in NP.

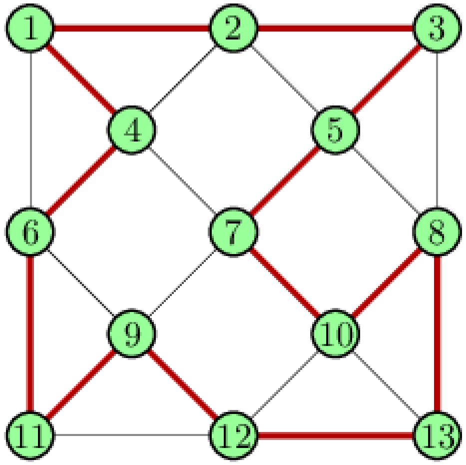
$$\begin{aligned} \mathrm{P(e)} \cdot \mathrm{P(f|e)} \quad = \quad & \mathrm{b}(e_1 \mid \mathrm{boundary}) \cdot \mathrm{b}(\mathrm{boundary} \mid e_l) \cdot \prod_{i=2}^{l} \mathrm{b}(e_i|e_{i-1}) \\ & \cdot\ \epsilon(m|l)\ \frac{1}{l^m} \prod_{j=1}^{m} \sum_{i=1}^{l} \mathrm{s}(f_j|e_i) \end{aligned}$$

All quantities on RHS are known. We simply calculate *P(e/f)*, and compare it to k

DECODING IS NP-HARD

By far, the most common way of proving a problem p as NP–hard is by reducing a known NP–hard problem to p. We will also use this approach here.

Consider the Hamiltonian Circuit problem (HC)



We will reduce each instance of HC to an instance of Decoding

REDUCTION FROM HC

Hamiltonian Circuit

Input
Graph G: vertices 0,1,..n;
Edge list E

Output

One bit, indicating if a Hamiltonian circuit exists or not

Decoding

Input
French string, f (length m)
Bigram Language model, b
Length probability table, ϵ
Channel model, s
Threshold, k in [0,1]

Output

One bit indicating if there is a source string with $P(e) \cdot P(f|e) > k$

REDUCTION FROM HC

For each vertex, 'create' a French word in the vocabulary

Vertices 1,2,…n $\longrightarrow$ French words f_1, f_2, f_3, … f_n

Create an English vocabulary with length n+1, containing extra word e_0. e_0 will work as the boundary word.

English words $\mathbf{e_0}$, e_1, e_2, e_3, … e_n

REDUCTION FROM HC

Create channel model tables as

$$s(f_j|e_i) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{otherwise} \end{cases}$$

$$\epsilon(m|l) = \begin{cases} 1 & \text{if } l = m \\ 0 & \text{otherwise} \end{cases}$$

What are the consequences of this?

All decodings, e, of f_1– f_n will contain **all** the words e_1, e_2, e_3, … e_n in **some order**. *Why?*

REDUCTION FROM HC

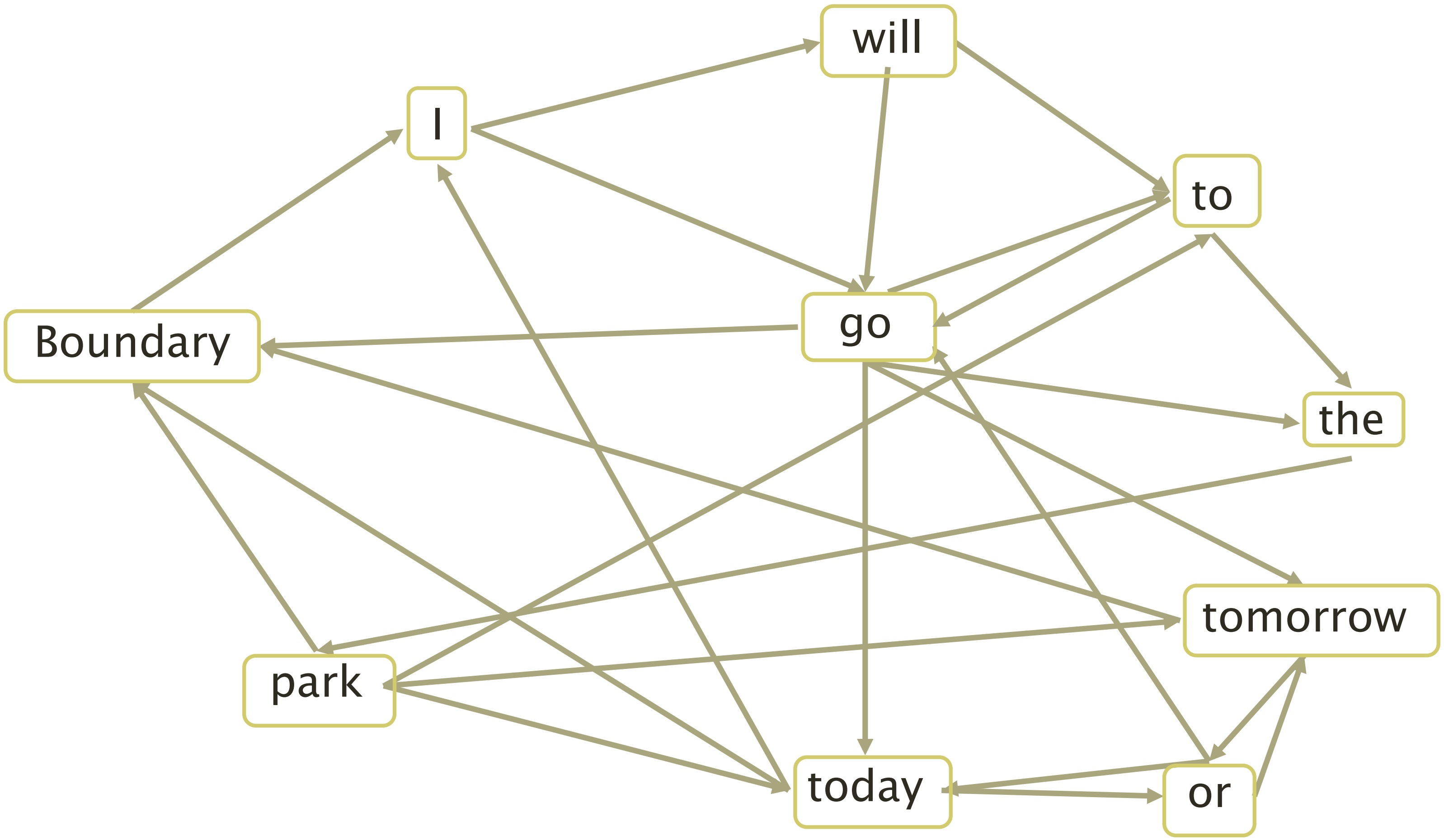
Now, create the language model, b, as:

$$\mathrm{b}(e_j|e_i) = \begin{cases} 1/\mathrm{n} & \text{if graph G contains an edge from vertex } i \text{ to vertex } j \\ 0 & \text{otherwise} \end{cases}$$

Now, set k = 0

Now our construction is complete. Let's see some intuition behind why it works.

REDUCTION FROM HC



Can you spot the Hamiltonian cycles?

REDUCTION FROM HC

Proof (YES)

So if Decoding returns YES, there must exist some string e with both P(e) and P(f|e) nonzero.

For P(f|e) non-zero, e must contain all words e_1, e_2, e_3, … e_n in some order

For P(e) is nonzero for a sentence, then every bigram in e must have nonzero probability.

Now this sentence corresponds to a path in the graph. Since each word or vertex is unique, and the sentence starts and ends at the boundary, this must be a Hamiltonian Cycle!

REDUCTION FROM HC

Proof (NO)

Since $k = 0$, ANY string e with non-zero probability, if it exists, will return a *YES* answer.

That is, for the decoding to return *NO*, ALL strings, e, include at least one zero value in the computation of either $P(e)$ or $P(f|e)$.

Now, consider the algorithm returns NO even if there is a possible circuit. This proposed circuit is simply an ordering of vertices. Given an ordering in G, we can construct a string e using the this, with $P(f|e) > 0$

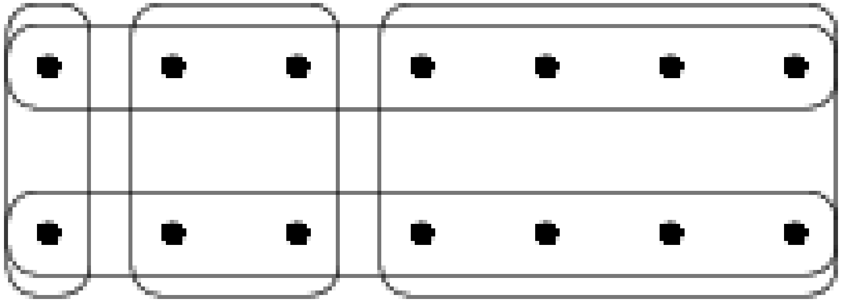
So it must be that $P(e) = 0$. But $P(e)$ is 0 iff the sentence does not form a loop. Therefore the given ordering is not a circuit at all!

Therefore Decoding returns YES if and only if a HC exists.

MINIMUM SET COVER

Given a set of elements (called the universe), a collection of sets whose union equals the universe, and an integer n,

The set cover problem is to identify a sub-collection with less than n subsets, such that union still equals the universe.



The decision problem is to state whether such a sub-collection exists or not

REDUCTION FROM MINIMUM SET COVER

Minimum Set Cover

Input

Finite Set S

Collection C of Subsets of set S

Integer n

Output

One bit, indicating if a C contain a cover of S of size <=n exists or not

Decoding

Input

French string, f (length m)

Bigram Language model, b

Length probability table, ϵ

Channel model, s

Threshold, k in [0,1]

Output

One bit indicating if there is a source string with $P(e) \cdot P(f|e) > k$

REDUCTION FROM MINIMUM SET COVER

For each subset in C , create a source word e_1 and let g_1 be the size of that subset

Subsets ,1,2,...m $\longrightarrow$ English words e_1, e_2, e_3, … e_m

Create table $b(e_i \mid e_j)$ with values set uniformly to the reciprocal of the number of subsets in C

For all i,j<m ,$b(e_i \mid e_j) = 1 / |\text{number of subsets in C}|$

For each element in S, create a French word f_1

Set elements ,1,2,...m $\longrightarrow$ French words f_1, f_2, f_3, … f_n

REDUCTION FROM MINIMUM SET COVER

Create channel model tables as

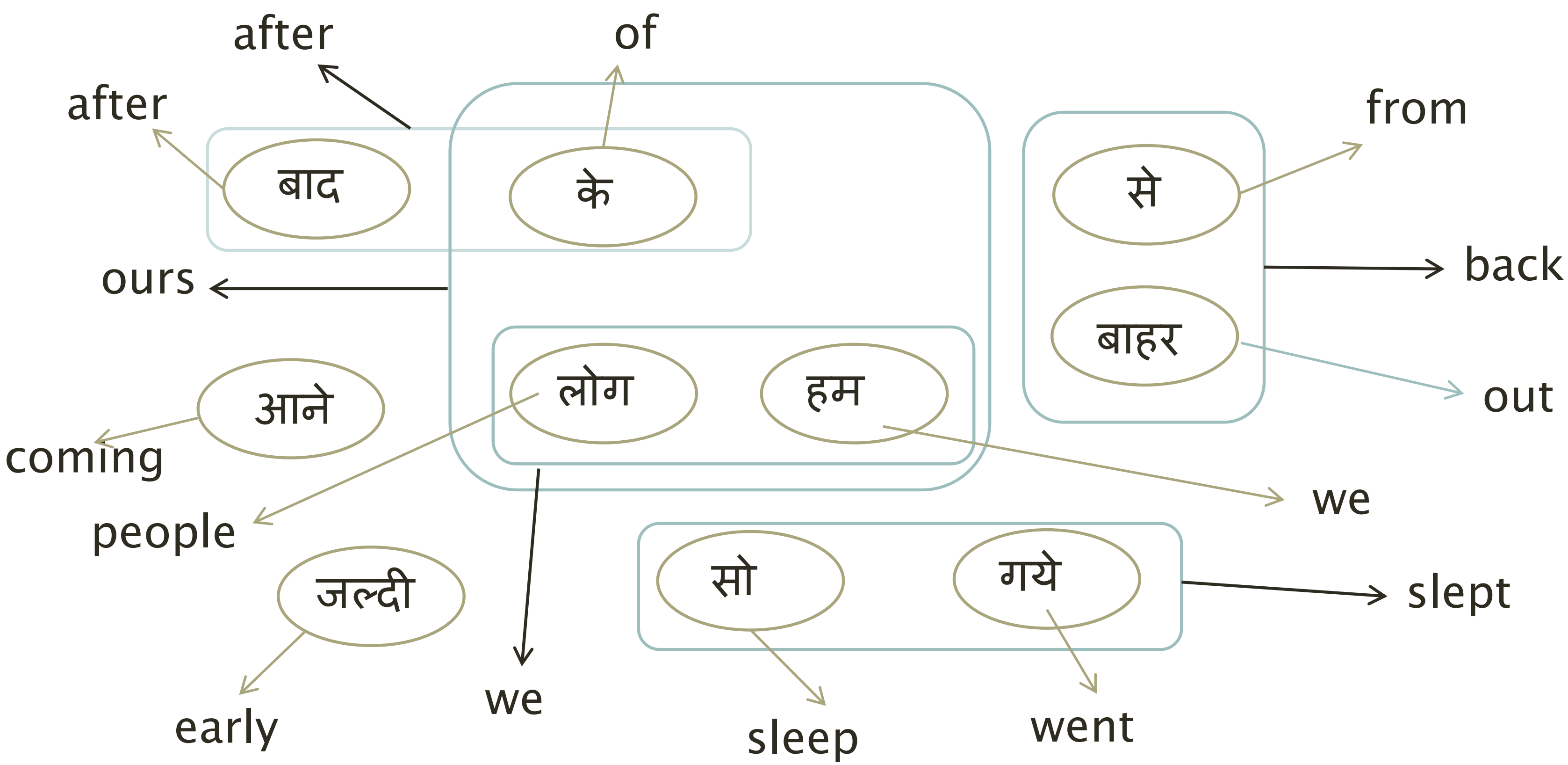
$$s(f_j|e_i) = \begin{cases} 1/g_i & \text{if the element in S corresponding to } f_j \text{ is also in the subset} \\ & \text{corresponding to } e_i \\ 0 & \text{otherwise} \end{cases}$$

$$\epsilon(m|l) = \begin{cases} 1 & \text{if } l \leq n \\ 0 & \text{otherwise} \end{cases}$$

Now, set k = 0

The parallel construction is complete.

REDUCTION FROM MINIMUM SET COVER



Selecting a concise set of source words

REDUCTION FROM MINIMUM SET COVER

Proof (YES)

if Decoding returns YES, there must exist some string e with

$P(e) . P(f|e) > 0$

$P(f|e) > 0$, if e must n or fewer words by the $\in$ table

For all $s(f_i|e_j) > 0$ tells us that every word in f_i is covered by at least one English word in e.

This is a set cover with #subsets less than n

REDUCTION FROM MINIMUM SET COVER

Proof (NO)

if Decoding returns NO, for all string e, $P(e) . P(f|e) = 0$

Since “table b” are all no zeroes , every e has $P(f|e) = 0$

$P(f|e) = 0$, iff

1) the length if e exceeds n

or

2) For some case, $s(f|e) = 0$, if f_i is left uncovered by the words in e.

So no set cover exists

CONCLUSIONS

So should we give up?

- NP hardness and NP completeness only talk about worst case time complexities. In practice, we may be able to get better run times on average cases.
- Often, heuristics are devised for NP hard problems to calculate ‘almost optimal’ solutions, with probabilistic guarantees on performance.
- Notably, Decoding transforms nicely into the Traveling Salesman Problem, for which excellent heuristics are known
- Other workaround techniques include:
  - Discarding unlikely alignments
  - Pre-processing texts to get likely alignment
  - Using a different channel model

THANK YOU