Decoding for SMT

Wilker Aziz

University of Sheffield
Sheffield, UK
w.aziz@sheffield.ac.uk

September 10, 2014
About this talk

This talk is not

- a review of beam search, cube pruning or any specific decoding algorithm
About this talk

This talk is not

▶ a review of beam search, cube pruning or any specific decoding algorithm

This talk is about

▶ understanding what makes decoding difficult
For starters

Let’s think of decoding as referring to an inference task
- making predictions
  - decisions in a highly combinatorial space of possibilities
For starters

Let’s think of decoding as referring to an inference task
  ▶ making predictions
      ▶ decisions in a highly combinatorial space of possibilities

Goals
For starters

Let’s think of decoding as referring to an inference task

- making predictions
  - decisions in a highly combinatorial space of possibilities

Goals

1. characterise the space of solutions
   (discuss tractability issues)
For starters

Let’s think of decoding as referring to an inference task
  ▶ making predictions
    ▶ decisions in a highly combinatorial space of possibilities

Goals

1. characterise the space of solutions
   (discuss tractability issues)
2. understand the impact of parameterisation
For starters

Let’s think of decoding as referring to an inference task
  ▶ making predictions
    ▶ decisions in a highly combinatorial space of possibilities

Goals

1. characterise the space of solutions
   (discuss tractability issues)
2. understand the impact of parameterisation
3. survey decoding techniques
Task

Translate a source text (e.g. sentence)

Examples:

*um conto de duas cidades* → *a tale of two cities*

*nosso amigo comum* → *our mutual friend*

*a loja de antiguidades* → *the old curiosity shop*

*o grill da lareira* → *the cricket on the hearth*
Model of translational equivalences

Defines the space of possible translations

- think of it as a recipe to generate translations
  [Lopez, 2008]
Model of translational equivalences

Defines the space of possible translations

▸ think of it as a recipe to generate translations
  [Lopez, 2008]

Example:

▸ a word replacement model
Model of translational equivalences

Defines the space of possible translations

- think of it as a recipe to generate translations
  [Lopez, 2008]

Example:

- a word replacement model
- operates in monotone left-to-right order
Model of translational equivalences

Defines the space of possible translations

▶ think of it as a recipe to generate translations
  [Lopez, 2008]

Example:

▶ a word replacement model
▶ operates in monotone left-to-right order
▶ with no insertions or deletions
Model of translational equivalences

Defines the space of possible translations

▶ think of it as a recipe to generate translations
  [Lopez, 2008]

Example:

▶ a word replacement model
▶ operates in monotone left-to-right order
▶ with no insertions or deletions
▶ constrained to known word-to-word bilingual mappings (rule set)
Monotone word-by-word translation: solutions

Source:  *um conto de duas cidades*

Translation rules

1. *um*  \( \{a, \text{some, one}\} \)
2. *conto*  \( \{\text{tale, story, narrative, novella}\} \)
3. *de*  \( \{\text{of, from, 's}\} \)
4. *duas*  \( \{\text{two, couple}\} \)
5. *cidades*  \( \{\text{cities, towns, villages}\} \)

---

1 Unrealistically simple
um conto de duas cidades
Monotone word-by-word translation: solutions

um \{ a, some, one \}
conto \{ tale, story, narrative, novella \}
de \{ of, from, 's \}
duas \{ two, couple \}
cidades \{ cities, towns, villages \}

*um conto de duas cidades*
a tale of two cities
Monotone word-by-word translation: solutions

um conto de duas cidades
a tale of two cities
a tale of two towns
Monotone word-by-word translation: solutions

*um* \{ a, some, one \}

*conto* \{ tale, story, narrative, novella \}

*de* \{ of, from, 's \}

*duas* \{ two, couple \}

*cidades* \{ cities, towns, villages \}

*um conto de duas cidades*

*a tale of two cities*

*a tale of two towns*

*a tale of two villages*
Monotone word-by-word translation: solutions

**um conto de duas cidades**

a tale of two cities

a tale of two **towns**

a tale of two **villages**

a tale of **couple cities**

**um, some, one**

**tale, story, narrative, novella**

**of, from, 's**

**two, couple**

**cities, towns, villages**
Monotone word-by-word translation: solutions

*um conto de duas cidades*

a tale of two cities
a tale of two **towns**
a tale of two **villages**
a tale of **couple cities**
a tale of couple **towns**
Monotone word-by-word translation: solutions

um conto de duas cidades
a tale of two cities
a tale of two towns
a tale of two villages
a tale of couple cities
a tale of couple towns
...

This can go very far :(
Monotone word-by-word translation: complexity

Say

- the input has $I$ words
- we know at most $t$ translation options per source word
Monotone word-by-word translation: complexity

Say

- the input has $I$ words
- we know at most $t$ translation options per source word

This makes $O(t^I)$ solutions
Monotone word-by-word translation: complexity

Say
- the input has $I$ words
- we know at most $t$ translation options per source word

This makes $O(t^I)$ solutions

Note
- WMT14’s shared task: $I = 40$ on average
- last I checked Moses default was $t = 100$
  (for a more complex model)
- silly monotone word replacement model: $10^{80}$ solutions
Space of solutions as intersection/composition

0  um  1  conto  2  de  3  duas  4  cidades

0  um  1  conto  2  de  3  duas  4  cidades

0  um  1  conto  2  de  3  duas  4  cidades

0  um  1  conto  2  de  3  duas  4  cidades
Space of solutions as intersection/composition
Space of solutions as intersection/composition

0 \rightarrow um^* \rightarrow 1 \rightarrow conto \rightarrow 2 \rightarrow de \rightarrow 3 \rightarrow duas \rightarrow 4 \rightarrow cidades \rightarrow 5

\rightarrow 0, a \rightarrow 1, a \rightarrow 2, a \rightarrow 3, a \rightarrow 4, a \rightarrow 5, a

\rightarrow a

Space of solutions as intersection/composition
Space of solutions as intersection/composition

0 $\xrightarrow{um^*}$ 1 $\xrightarrow{conto}$ 2 $\xrightarrow{de}$ 3 $\xrightarrow{duas}$ 4 $\xrightarrow{cidades}$ 5

0, $a$ $\xleftarrow{\text{some}}$ 1, $a$

2, $a$ $\xleftarrow{\text{duas}}$ 3, $a$

4, $a$ $\xleftarrow{\text{cidades}}$ 5, $a$

$um:a \checkmark$

$um:some \checkmark$

$um:one \leftarrow$

conto:tale

conto:story

conto:narrative

duo:novella

de:of

de:from

de:'s

duo:two

duo:couple

cidades:cities

cidades:towns

cidades:villages

\[\rightarrow a\]
Space of solutions as intersection/composition

\[
\begin{array}{cccccc}
0 & \phantom{0} & \text{um} & \phantom{0} & 1 & \text{conto}^* \\
\text{conto} & \phantom{0} & \text{de} & \phantom{0} & 3 & \text{duas} \\
\text{duas} & \phantom{0} & \phantom{0} & \phantom{0} & 4 & \text{cidades} \\
\text{cidades} & \phantom{0} & \phantom{0} & \phantom{0} & \phantom{0} & \phantom{0} & 5 \\
\end{array}
\]
Space of solutions as intersection/composition

0 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5

um: a ✓
um: some ✓
um: one ✓
conto: tale ✓
conto: story ←
conto: narrative
conto: novella
de: of
de: from
de: 's
duas: two
duas: couple
cidades: cities
cidades: towns
cidades: villages

0, a \rightarrow 1, a \rightarrow 2, a

a

some

one

tale

story

3, a

4, a

5, a

a
Space of solutions as intersection/composition

um:a ✓
um:some ✓
um:one ✓
conto:tale ✓
conto:story ✓
conto:narrative ←
conto:interest
de:of
de:from
de:’s
duas:two
duas:couple
cidades:cities
cidades:towns
cidades:villages
Space of solutions as intersection/composition

um: a ✓
um: some ✓
um: one ✓
conto: tale ✓
conto: story ✓
conto: narrative ✓
conto: novella ←
de: of
de: from
de: 's
duas: two
duas: couple
cidades: cities
cidades: towns
cidades: villages
Space of solutions as intersection/composition

um: a ✓
um: some ✓
um: one ✓
conto: tale ✓
conto: story ✓
conto: narrative ✓
conto: novella ✓
de: of ←
de: from
de: 's
duas: two
duas: couple
cidades: cities
cidades: towns
cidades: villages
Space of solutions as intersection/composition

0 \(\xrightarrow{um} 1 \xrightarrow{conto} 2 \xrightarrow{de^*} 3 \xrightarrow{duas} 4 \xrightarrow{cidades} 5\)

0, \(a\) \xrightarrow{a} 1, \(a\) \xrightarrow{tale\; story\; narrative\; novella} 2, \(a\) \xrightarrow{of\; from} 3, \(a\) \xrightarrow{} 4, \(a\) \xrightarrow{} 5, \(a\) \xrightarrow{a}\n
\text{um:a \checkmark}
\text{um:some \checkmark}
\text{um:one \checkmark}
\text{conto:tale \checkmark}
\text{conto:story \checkmark}
\text{conto:narrative \checkmark}
\text{conto:novella \checkmark}
\text{de:of \checkmark}
\text{de:from \leftarrow}
\text{de:'s}
\text{duas:two}
\text{duas:couple}
\text{cidades:cities}
\text{cidades:towns}
\text{cidades:villages}
Space of solutions as intersection/composition
Space of solutions as intersection/composition

0 \rightarrow \text{um} \rightarrow 1 \rightarrow \text{conto} \rightarrow 2 \rightarrow \text{de} \rightarrow 3 \rightarrow \text{duas}^* \rightarrow 4 \rightarrow \text{cidades} \rightarrow 5

0, a \rightarrow a \rightarrow \text{some} \rightarrow \text{one} \rightarrow \text{conto} \rightarrow \text{tale} \rightarrow \text{story} \rightarrow \text{narrative} \rightarrow \text{of} \rightarrow \text{from} \rightarrow \text{'s} \rightarrow \text{two} \rightarrow 5, a \rightarrow a
Space of solutions as intersection/composition

0 \rightarrow um \rightarrow 1 \rightarrow conto \rightarrow 2 \rightarrow de \rightarrow 3 \rightarrow duas^* \rightarrow cidades \rightarrow 5

0, a \rightarrow a \rightarrow some \rightarrow 1, a \rightarrow tale \rightarrow story \rightarrow 2, a \rightarrow of \rightarrow from \rightarrow 3, a \rightarrow two \rightarrow couple \rightarrow 4, a \rightarrow two \rightarrow the \rightarrow couple \rightarrow 5, a

um:a ✓
um:some ✓
um:one ✓
conto:tale ✓
conto:story ✓
conto:narrative ✓
conto:novella ✓
de:of ✓
de:from ✓
de:'s ✓
duas:two ✓
duas:couple ←
cidades:cities
cidades:towns
cidades:villages

a
Space of solutions as intersection/composition

0 → um → 1 → conto → 2 → de → 3 → duas → 4 → cidades* → 5

0, a → a → some → 1, a → story → 2, a → of → 3, a → couple → 4, a → cities → 5, a

um: a ✓
um: some ✓
um: one ✓
conto: tale ✓
conto: story ✓
conto: narrative ✓
conto: novella ✓
de: of ✓
de: from ✓
de: 's ✓
duas: two ✓
duas: couple ✓
cidades: cities ← cidades: towns
cidades: villages
Space of solutions as intersection/composition

Graph showing transitions between terms:
- 0 → um
- um → 1
- 1 → conto
- conto → 2
- 2 → de
- de → 3
- 3 → duas
- duas → 4
- 4 → cidades
- cidades → 5
- 5 → a

Terms:
- um: a ✓
- um: some ✓
- um: one ✓
- conto: tale ✓
- conto: story ✓
- conto: narrative ✓
- conto: novella ✓
- de: of ✓
- de: from ✓
- de: 's ✓
- duas: two ✓
- duas: couple ✓
- cidades: cities ✓
- cidades: towns ←
- cidades: villages
Space of solutions as intersection/composition

\[ \text{um: a} \checkmark \quad \text{um: some} \checkmark \quad \text{um: one} \checkmark \quad \text{conto: tale} \checkmark \quad \text{conto: story} \checkmark \quad \text{conto: narrative} \checkmark \quad \text{conto: novella} \checkmark \quad \text{de: of} \checkmark \quad \text{de: from} \checkmark \quad \text{de: 's} \checkmark \quad \text{duas: two} \checkmark \quad \text{duas: couple} \checkmark \quad \text{cidades: cities} \checkmark \quad \text{cidades: towns} \checkmark \quad \text{cidades: villages} \leftarrow \]
Space of solutions as intersection/composition

\[ 3 \times 4 \times 3 \times 2 \times 3 = 216 \text{ solutions} \]

- 6 states
- \( 3 + 4 + 3 + 2 + 3 = 15 \) transitions
Packing solutions with finite-state automata

Same $O(t^I)$ solutions using

- $O(I)$ states
- $O(tI)$ transitions
Recap 1
Recap 1

Model of translational equivalences
- defines the space of possible sentence pairs
- conveniently decomposes into smaller bilingual mappings
Recap 1

Model of translational equivalences
  ▶ defines the space of possible sentence pairs
  ▶ conveniently decomposes into smaller bilingual mappings

Monotone word replacement model
  ▶ easy to represent using finite-state transducers
Recap 1

Model of translational equivalences
- defines the space of possible sentence pairs
- conveniently decomposes into smaller bilingual mappings

Monotone word replacement model
- easy to represent using finite-state transducers
- set of translations given by composition
Recap 1

Model of translational equivalences
- defines the space of possible sentence pairs
- conveniently decomposes into smaller bilingual mappings

Monotone word replacement model
- easy to represent using finite-state transducers
- set of translations given by composition
- exponential number of solutions in linear space
Recap 1

Model of translational equivalences
▶ defines the space of possible sentence pairs
▶ conveniently decomposes into smaller bilingual mappings

Monotone word replacement model
▶ easy to represent using finite-state transducers
▶ set of translations given by composition
▶ exponential number of solutions in linear space
▶ translates infinitely many sentences
Recap 1

Model of translational equivalences
- defines the space of possible sentence pairs
- conveniently decomposes into smaller bilingual mappings

Monotone word replacement model
- easy to represent using finite-state transducers
- set of translations given by composition
- exponential number of solutions in linear space
- translates infinitely many sentences
  but not nearly enough interesting cases!
Monotone word-by-word translation: fail!

- *nosso*: \{our, ours\}
- *amigo*: \{friend, mate\}
- *comum*: \{ordinary, common, usual, mutual\}
Monotone word-by-word translation: fail!

- nossos: {our, ours}
- amigo: {friend, mate}
- comum: {ordinary, common, usual, mutual}
Monotone word-by-word translation: fail!

\[
\begin{align*}
\text{nosso} & \quad \{\text{our, ours}\} \\
\text{amigo} & \quad \{\text{friend, mate}\} \\
\text{comum} & \quad \{\text{ordinary, common, usual, mutual}\}
\end{align*}
\]
Monotone word-by-word translation: fail!

- **nosso** \{our, ours\}
- **amigo** \{friend, mate\}
- **comum** \{ordinary, common, usual, mutual\}
Monotone word-by-word translation: fail!

\[
\begin{align*}
nosso & \rightarrow \{ \text{our, ours} \} \\
amigo & \rightarrow \{ \text{friend, mate} \} \\
comum & \rightarrow \{ \text{ordinary, common, usual, mutual} \}
\end{align*}
\]

We simply cannot obtain a correct translation

our mutual friend
Our model of translational equivalences assumes monotonicity

- a word replacement model
- operates in **monotone** left-to-right order
- with no insertions or deletions
- constrained to known word-to-word bilingual mappings (rule set)
Reordering

Not anymore!

- a word replacement model
- operates in arbitrary order
- with no insertions or deletions
- constrained to known word-to-word bilingual mappings (rule set)
Translating arbitrary permutations

*nosso amigo comum*
Translating arbitrary permutations

*nosso amigo comum*

0 → 1, ours ~ our
1 → 2, friend ~ mate
2 → 3, ordinary ~ usual
3 → 0, mutual

*amigo nosso comum*

0 → 1, friend ~ mate
1 → 2, ours ~ our
2 → 3, ordinary ~ usual
3 → 0, mutual
Translating arbitrary permutations

**nosso amigo comum**

0 → 1 (ours) → 2 (friend) → 3 (ordinary)
1 → 2 (ours) → 3 (friend)
2 → 3 (ours) → 0 (friend)
3 → 0 (ours) → 1 (friend)

**amigo nosso comum**

0 → 1 (mate) → 2 (ordinary) → 3 (usual)
1 → 2 (mate) → 3 (usual)
2 → 3 (mate) → 0 (usual)
3 → 0 (mate) → 1 (usual)

**nosso comum amigo**

0 → 1 (ours) → 2 (ordinary) → 3 (usual)
1 → 2 (ours) → 3 (usual)
2 → 3 (ours) → 0 (usual)
3 → 0 (ours) → 1 (usual)
Translating arbitrary permutations

**nosso amigo comum**

0 → ours → 1 → mate → 2 → usual → 3

**amigo nosso comum**

0 → friend → 1 → ours → 2 → usual → 3

**nosso comum amigo**

0 → ours → 1 → ordinary → 2 → usual → 3

**comum nosso amigo**

0 → ordinary → 1 → ours → 2 → usual → 3
Translating arbitrary permutations

nosso amigo comum

amigo nosso comum

nosso comum amigo

comum nosso amigo

amigo comum nosso
Translating arbitrary permutations

nosso amigo comum

amigo nosso comum

nosso comum amigo

comum nosso amigo

amigo comum nosso

comum amigo nosso
Translating arbitrary permutations

\[ 3! = 3 \times 2 \times 1 = 6 \text{ permutations} \]
Translating arbitrary permutations

**nosso amigo comum**

- 0 → ours → our
- 1 → friend → mate
- 2 → ordinary → common
- 3 → usual → mutual

**amigo nosso comum**

- 0 → friend → ours
- 1 → ours → our
- 2 → ordinary → common
- 3 → usual → mutual

**nosso comum amigo**

- 0 → ours → our
- 1 → ordinary → common
- 2 → usual → mutual
- 3 → friend 

**comum nosso amigo**

- 0 → our → ours
- 1 → common → usual
- 2 → mutual
- 3 → friend

**amigo comum nosso**

- 0 → friend → mate
- 1 → ordinary → common
- 2 → usual → mutual
- 3 → ours

**comum amigo nosso**

- 0 → ordinary → common
- 1 → usual → mutual
- 2 → friend 
- 3 → ours

Each has $2 \times 2 \times 4 = 16$ translations
Translating arbitrary permutations

amounting to $6 \times 16 = 96$ solutions
Translating arbitrary permutations

**nosso amigo comum**

0 1 2 3

- ours -> our
- friend -> mate
- ordinary -> common
- usual -> mutual

**amigo nosso comum**

0 1 2 3

- friend -> ours
- ours -> ordinary
- common -> usual
- usual -> mutual

**nosso comum amigo**

0 1 2 3

- ours -> our
- common -> usual
- usual -> mutual
- mutual -> friend

**comum nosso amigo**

0 1 2 3

- our -> ours
- common -> usual
- usual -> mutual
- mutual -> friend

**amigo comum nosso**

0 1 2 3

- friend -> mate
- ordinary -> common
- usual -> mutual
- mutual -> ours

**comum amigo nosso**

0 1 2 3

- ordinary -> common
- usual -> mutual
- mutual -> friend
- friend -> ours

$I!$ permutations $\times t^I$ translations
Packing permutations
Packing permutations
Packing permutations

0 → 1
1 → 2
2 → 3

nosso
amigo
comum

comum
Packing permutations
Packing permutations

Diagram showing permutations with transitions labeled 'amigo', 'comum', and 'nosso'.
Packing permutations
Packing permutations
Packing permutations

Powerset (all subsets) of \( \{1, 2, \ldots, I\} \)
- \( 2^I \) subsets
  - think of a vector of \( I \) bits ;)

Lattice
- \( O(2^I) \) states
- \( O(I \times 2^I) \) transitions
Word replacement with unconstrained reordering

Source: *nosso amigo comum*
Word replacement with unconstrained reordering

Source: *nosso amigo comum*

1. arbitrary permutations: $O(2^I)$ states
Word replacement with unconstrained reordering

Source: *nosso amigo comum*

1. arbitrary permutations: $O(2^I)$ states
2. intersection with the rule set: $O(tI2^I)$ transitions
Problem!

Before we even discuss a parameterisation of the model we already ran into a tractability issue!
Problem!

Before we even discuss a parameterisation of the model we already ran into a tractability issue!

- NP-complete [Knight, 1999]
- generalised TSP
Before we even discuss a parameterisation of the model we already ran into a tractability issue!

▶ NP-complete [Knight, 1999]
▶ generalised TSP

Direction

▶ is it sensible to consider the space of all permutations?
Problem!

Before we even discuss a parameterisation of the model we already ran into a tractability issue!

- NP-complete [Knight, 1999]
- generalised TSP

Direction

- is it sensible to consider the space of all permutations?

Solution
Problem!

Before we even discuss a parameterisation of the model we already ran into a tractability issue!

- NP-complete [Knight, 1999]
- generalised TSP

Direction

- is it sensible to consider the space of all permutations?

Solution

- constrain reordering :D
Problem!

Before we even discuss a parameterisation of the model we already ran into a tractability issue!

- NP-complete [Knight, 1999]
- generalised TSP

Direction

- is it sensible to consider the space of all permutations?

Solution

- constrain reordering :D
- 0.o but how?
Ad-hoc distortion limit

Several flavours of distortion limit [Lopez, 2009]
Ad-hoc distortion limit

Several flavours of distortion limit [Lopez, 2009]

- limit reordering as a function of the number of skipped words
Ad-hoc distortion limit

Several flavours of distortion limit [Lopez, 2009]

- limit reordering as a function of the number of skipped words

Moses allows reordering within a window of length $d$

- starting from the leftmost uncovered word
Suppose $d = 2$ and $I = 3$
Suppose $d = 2$ and $I = 3$
Suppose $d = 2$ and $I = 3$
Suppose $d = 2$ and $I = 3$
Suppose \( d = 2 \) and \( I = 3 \)
Suppose $d = 2$ and $I = 3$
Suppose $d = 2$ and $I = 3$
Suppose $d = 2$ and $I = 3$
Suppose $d = 2$ and $I = 3$
Suppose $d = 2$ and $I = 3$ (e.g. *nosso amigo comum*)

\[
\begin{align*}
\{1, 2\} & \xrightarrow{\text{nosso}} \{2, 3\} \\
\{2, 3\} & \xrightarrow{\text{comum}} \{2, \emptyset\} \\
\{1, \emptyset\} & \xrightarrow{\text{amigo}} \{2, 3\} \\
\{1, \emptyset\} & \xrightarrow{\text{amigo}} \{3\} \\
\{2, 3\} & \xrightarrow{\text{amigo}} \{3\} \\
\{3\} & \xrightarrow{\text{comum}} \emptyset
\end{align*}
\]
Word replacement with reordering constrained by WL2

Complexity: $O(I2^{d-1})$ states
Complexity: $O(tI2^{d-1})$ transitions
Ad-hoc distortion limit: expressiveness

Arbitrarily limit reordering to a fixed-length window

Arbitrarily limit reordering to a fixed-length window
Ad-hoc distortion limit: expressiveness

Arbitrarily limit reordering to a fixed-length window
▶ convenient (linear complexity), but
Ad-hoc distortion limit: expressiveness

Arbitrarily limit reordering to a fixed-length window

▶ convenient (linear complexity), but
▶ what about languages with very different syntax?
  e.g. OV vs VO, head-initial vs head-final
Ad-hoc distortion limit: expressiveness

Arbitrarily limit reordering to a fixed-length window

- convenient (linear complexity), but
- what about languages with very different syntax? e.g. OV vs VO, head-initial vs head-final
- can we do better?
Binary permutations

Consider a sentence such that $I = 4$
let’s look at binary bracketing trees for this sentence
Consider a sentence such that \( I = 4 \)

let’s look at binary bracketing trees for this sentence
Binary permutations

Consider a sentence such that $I = 4$
let's look at binary bracketing trees for this sentence

Binary permutations

$(((1 \ 2)3)4)$ 1 2 3 4
Consider a sentence such that \( I = 4 \)
let’s look at binary bracketing trees for this sentence

Binary permutations

\[
(((1\ 2)3)4) \quad 1\ 2\ 3\ 4
\]

\[
((1\ 2)(3)4) \quad 2\ 1\ 3\ 4
\]
Consider a sentence such that \( I = 4 \)

let’s look at binary bracketing trees for this sentence

Binary permutations

\[
(((1 \ 2)3)4) \ 1 \ 2 \ 3 \ 4 \\
((\langle1 \ 2\rangle3)4) \ 2 \ 1 \ 3 \ 4 \\
(\langle\langle1 \ 2\rangle3\rangle4) \ 3 \ 1 \ 2 \ 4
\]
Consider a sentence such that \( I = 4 \)

let’s look at binary bracketing trees for this sentence

Binary permutations

\[
\begin{align*}
((1 \ 2)3)4 & \quad 1 \ 2 \ 3 \ 4 \\
((1 \ 2)34) & \quad 2 \ 1 \ 3 \ 4 \\
(\langle(1 \ 2)3\rangle4) & \quad 3 \ 1 \ 2 \ 4 \\
(\langle\langle1 \ 2\rangle3\rangle4) & \quad 3 \ 2 \ 1 \ 4
\end{align*}
\]
Binary permutations

Consider a sentence such that $I = 4$

let’s look at binary bracketing trees for this sentence

Binary permutations

- (((1 2)3)4) 1 2 3 4
- ((⟨1 2⟩)3)4) 2 1 3 4
- ⟨⟨1 2⟩3⟩4) 3 1 2 4
- ⟨⟨⟨1 2⟩3⟩⟩4) 3 2 1 4

...
Consider a sentence such that $I = 4$
let’s look at binary bracketing trees for this sentence

```
1 2 3 4
1 2 3 4
1 2 3 4
1 2 3 4
```

Binary permutations

- $((1(2 3))4)$   1 2 3 4
- $((1\langle 2 3 \rangle)4)$  1 3 2 4
- $(\langle 1(2 3)\rangle 4)$  2 3 1 4
- $(\langle 1\langle 2 3 \rangle\rangle 4)$  3 2 1 4

...
Consider a sentence such that $I = 4$
let’s look at binary bracketing trees for this sentence

Binary permutations

\[
\begin{align*}
( (1 \ 2)(3 \ 4)) & \quad 1 \ 2 \ 3 \ 4 \\
(\langle 1 \ 2\rangle(3 \ 4)) & \quad 2 \ 1 \ 3 \ 4 \\
(\langle 1 \ 2\rangle\langle 3 \ 4\rangle) & \quad 2 \ 1 \ 4 \ 3 \\
((1 \ 2)(3 \ 4)) & \quad 1 \ 2 \ 4 \ 3 \\
((1 \ 2)(3 \ 4)) & \quad 1 \ 2 \ 4 \ 3 \\
\end{align*}
\]
Binary permutations

Consider a sentence such that $I = 4$
let’s look at binary bracketing trees for this sentence

Binary permutations

$(1((2\ 3)4))$ 1 2 3 4
$(1(⟨2\ 3⟩4))$ 1 3 2 4
$(1⟨(2\ 3)4⟩)$ 1 4 2 3
$(1⟨⟨2\ 3⟩4⟩)$ 1 4 3 2
...

1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4
Binary permutations

Consider a sentence such that $I = 4$

let's look at binary bracketing trees for this sentence

Binary permutations

\[
\begin{align*}
(1(2(3 \ 4))) & \quad 1 \ 2 \ 3 \ 4 \\
(1(2\langle 3 \ 4\rangle)) & \quad 1 \ 2 \ 4 \ 3 \\
(1\langle 2(3 \ 4)\rangle) & \quad 1 \ 3 \ 4 \ 2 \\
(1\langle 2\langle 3 \ 4\rangle\rangle) & \quad 1 \ 4 \ 3 \ 2
\end{align*}
\]

...
Binary permutations

Consider a sentence such that $I = 4$

let’s look at binary bracketing trees for this sentence

Binary permutations

- constrains the space of permutations
Consider a sentence such that $I = 4$

let’s look at binary bracketing trees for this sentence

Binary permutations

- constrains the space of permutations
- crossing constraint
  - $3 \ 1 \ 4 \ 2 \ \times$
  - $2 \ 4 \ 1 \ 3 \ \times$
ITGs

Inversion Transduction Grammars (ITGs) [Wu, 1997]
Inversion Transduction Grammars (ITGs) [Wu, 1997]

- $X \rightarrow XX$
  
  direct order
Inversion Transduction Grammars (ITGs) [Wu, 1997]

- $X \rightarrow XX$
  - direct order
- $X \rightarrow \langle XX \rangle$
  - inverted order
Inversion Transduction Grammars (ITGs) [Wu, 1997]

- $X \rightarrow XX$
  direct order
- $X \rightarrow \langle XX \rangle$
  inverted order
- $X \rightarrow f/e$, where $(f, e) \in R$
  bilingual mappings
Parsing, intersection and hypergraphs

Source

0 \(\rightarrow\) 1 \(\rightarrow\) 2 \(\rightarrow\) 3

\(\text{nossa} \rightarrow \text{amigo} \rightarrow \text{comum}\)
Parsing, intersection and hypergraphs

Source

Grammar

$X \rightarrow XX$

$X \rightarrow \langle XX \rangle$

$X \rightarrow \text{nosso}$

$X \rightarrow \text{amigo}$

$X \rightarrow \text{comum}$
Parsing, intersection and hypergraphs

Source

Grammar
\[
X \rightarrow XX \\
X \rightarrow \langle XX \rangle \\
X \rightarrow \text{nosso} \\
X \rightarrow \text{amigo} \\
X \rightarrow \text{comum}
\]
Parsin, intersection and hypergraphs

Source

Grammar
\[ X \rightarrow XX \]
\[ X \rightarrow \langle XX \rangle \]
\[ X \rightarrow \text{nosso} \]
\[ X \rightarrow \text{amigo} \]
\[ X \rightarrow \text{comum} \]

\[ \text{X}_1 \]
Parsing, intersection and hypergraphs

Source

Grammar

\[ X \rightarrow XX \]
\[ X \rightarrow \langle XX \rangle \]
\[ X \rightarrow \text{nosso} \]
\[ X \rightarrow \text{amigo} \]
\[ X \rightarrow \text{comum} \]
Parsing, intersection and hypergraphs

Source

Grammar

X → XX
X → ⟨XX⟩
X → nosso
X → amigo
X → comum

X

nosso

1

amigo

2
comum

3

0

X₁

0

X₂

1

X₃

2

X

X

←
Parsing, intersection and hypergraphs

Source

Grammar

\[
\begin{align*}
X & \rightarrow XX \\
X & \rightarrow \langle XX \rangle \\
X & \rightarrow \text{nosso} \\
X & \rightarrow \text{amigo} \\
X & \rightarrow \text{comum}
\end{align*}
\]
Parsing, intersection and hypergraphs

Source

Grammar

X → XX
X → ⟨XX⟩
X → nosso
X → amigo
X → comum
Parsing, intersection and hypergraphs

Source

Grammar

\[ X \rightarrow XX \]
\[ X \rightarrow \langle XX \rangle \]
\[ X \rightarrow \text{nosso} \]
\[ X \rightarrow \text{amigo} \]
\[ X \rightarrow \text{comum} \]
Parsing, intersection and hypergraphs

Source

Grammar

\[ X \rightarrow XX \]
\[ X \rightarrow \langle XX \rangle \]
\[ X \rightarrow \text{nosso} \]
\[ X \rightarrow \text{amigo} \]
\[ X \rightarrow \text{comum} \]
Parsing, intersection and hypergraphs

Source

Grammar

\[
X \rightarrow XX \\
X \rightarrow \langle XX \rangle \\
X \rightarrow \text{nosso} \\
X \rightarrow \text{amigo} \\
X \rightarrow \text{comum}
\]
Parsing, intersection and hypergraphs

Source

\[ 0 \xrightarrow{noso} 1 \xrightarrow{amigo} 2 \xrightarrow{comum} 3 \]

Grammar

\[
\begin{align*}
X & \rightarrow XX \\
X & \rightarrow \langle XX \rangle \\
X & \rightarrow \text{noso} \\
X & \rightarrow \text{amigo} \\
X & \rightarrow \text{comum}
\end{align*}
\]
Parsing, intersection and hypergraphs

Source

\[ X \rightarrow XX \]
\[ X \rightarrow \langle XX \rangle \]
\[ X \rightarrow \text{nosso} \]
\[ X \rightarrow \text{amigo} \]
\[ X \rightarrow \text{comum} \]
Parsing, intersection and hypergraphs

Source

Grammar

\[ X \rightarrow XX \]
\[ X \rightarrow \langle XX \rangle \]
\[ X \rightarrow \text{nosso} \]
\[ X \rightarrow \text{amigo} \]
\[ X \rightarrow \text{comum} \]
Parsing, intersection and hypergraphs

Source

\[ 0 \rightarrow \text{nosso} \rightarrow 1 \rightarrow \text{amigo} \rightarrow 2 \rightarrow \text{comum} \rightarrow 3 \]

Grammar

\[
\begin{align*}
X & \rightarrow XX \\
X & \rightarrow \langle XX \rangle \\
X & \rightarrow \text{nosso} \\
X & \rightarrow \text{amigo} \\
X & \rightarrow \text{comum}
\end{align*}
\]
Parsing, intersection and hypergraphs

Source

0 \rightarrow nossos \rightarrow 1 \rightarrow amigo \rightarrow 2 \rightarrow comum \rightarrow 3

Grammar

X \rightarrow XX
X \rightarrow \langle XX \rangle
X \rightarrow nossos
X \rightarrow amigo
X \rightarrow comum

\( O(I^3) \) nodes
\( O(tI^3) \) edges
Example

\[(\text{n favourite} \langle \text{amigo comum} \rangle) \rightarrow \text{our mutual friend}\]
1. our first model of translational equivalences assumed
   monotonicity
1. our first model of translational equivalences assumed **monotonicity**
2. then we incorporated **unconstrained permutations** of the input
Recap 2

1. our first model of translational equivalences assumed **monotonicity**

2. then we incorporated **unconstrained permutations** of the input

3. to avoid NP-completeness, we constrained permutations using a **distortion limit**
Recap 2

1. our first model of translational equivalences assumed **monotonicity**
2. then we incorporated **unconstrained permutations** of the input
3. to avoid NP-completeness, we constrained permutations using a **distortion limit**
4. we can instead constrain permutations using an **ITG**
1. our first model of translational equivalences assumed **monotonicity**
2. then we incorporated **unconstrained permutations** of the input
3. to avoid NP-completeness, we constrained permutations using a **distortion limit**
4. we can instead constrain permutations using an **ITG**

But we still perform translation word-by-word with no insertion or deletion!
1-1 mappings: fail!

Source: o₁ grilo₂ da₃ lareira₄
Target: the₁ cricket₂ [on the]₃ hearth₄
Insertion and deletion

Implicitly modelled by moving from words to phrases
Insertion and deletion

Implicitly modelled by moving from words to phrases
  ▶ a phrase replacement model
Insertion and deletion

Implicitly modelled by moving from words to phrases

- a phrase replacement model
- operating with an ITG (or with a distortion limit)
Insertion and deletion

Implicitly modelled by moving from words to phrases

▶ a phrase replacement model
▶ operating with an ITG (or with a distortion limit)
▶ with no phrase-insertion or phrase-deletion
Insertion and deletion

Implicitly modelled by moving from words to phrases

- a phrase replacement model
- operating with an ITG (or with a distortion limit)
- with no phrase-insertion or phrase-deletion
- constrained to known phrase-to-phrase bilingual mappings (rule set)
Phrase mappings

Mappings of contiguous sequences of words
Phrase mappings

Mappings of contiguous sequences of words
  ▶ learnt directly (e.g. stochastic ITGs)

- a loja de antiguidades
  /old curiosity shop

- their words need not align monotonically
  which gives us a bit of reordering power as well ;)
Phrase mappings

Mappings of contiguous sequences of words

- learnt directly (e.g. stochastic ITGs)
- heuristically extracted from word-aligned data

- they might contain unaligned source words (deletions)
- they might contain unaligned target words (insertions)
- their words need not align monotonically

which gives us a bit of reordering power as well ;)

- e.g. a loja de antiguidades / old curiosity shop
Phrase mappings

Mappings of contiguous sequences of words

- learnt directly (e.g. stochastic ITGs)
- heuristically extracted from word-aligned data
- they might contain unaligned source words (deletions)

- they might contain unaligned target words (insertions)

Which gives us a bit of reordering power as well ;)

- e.g. a loja de antiguidades
  /old curiosity shop
Phrase mappings

Mappings of contiguous sequences of words
- learnt directly (e.g. stochastic ITGs)
- heuristically extracted from word-aligned data
- they might contain unaligned source words (deletions)
- they might contain unaligned target words (insertions)
Phrase mappings

Mappings of contiguous sequences of words

- learnt directly (e.g. stochastic ITGs)
- heuristically extracted from word-aligned data
- they might contain unaligned source words (deletions)
- they might contain unaligned target words (insertions)
- their words need not align monotonically
  which gives us a bit of reordering power as well ;)

a loja de antiguidades  /old curiosity shop
Phrase mappings

Mappings of contiguous sequences of words

- learnt directly (e.g. stochastic ITGs)
- heuristically extracted from word-aligned data
- they might contain unaligned source words (deletions)
- they might contain unaligned target words (insertions)
- their words need not align monotonically
  which gives us a bit of reordering power as well ;)

 e.g. *a loja de antiguidades*/*old curiosity shop*
Generalising the rule set (FST)

Rules
- o: \{the, a\}
- grilo: \{cricket, annoyance\}
- da: \{on the, of, from\}
- hearth: \{lareira\}
Generalising the rule set (FST)

Rules
- $o$ \{the, a\}
- $grilo$ \{cricket, annoyance\}
- $da$ \{on the, of, from\}
- $hearth$ \{lareira\}

Using FST
- each rule can be seen as a transducer
Generalising the rule set (FST)

Rules
- \( o \) \{the, a\}
- grilo \{cricket, annoyance\}
- da \{on the, of, from\}
- hearth \{lareira\}

Using FST
- each rule can be seen as a transducer
Generalising the rule set (FST)

Rules

- **o**: \{the, a\}
- **grilo**: \{cricket, annoyance\}
- **da**: \{on the, of, from\}
- **hearth**: \{lareira\}

Using FST

- each rule can be seen as a transducer
Generalising the rule set (FST)

Rules
- \(o\) \{the, a\}
- \(grilo\) \{cricket, annoyance\}
- \(da\) \{on the, of, from\}
- \(hearth\) \{lareira\}

Using FST
- each rule can be seen as a transducer
Generalising the rule set (FST)

Rules

- `o` {the, a}
- `grilo` {cricket, annoyance}
- `da` {on the, of, from}
- `hearth` {lareira}

Using FST

- each rule can be seen as a transducer
Generalising the rule set (FST)

Rules
- **o**: \{the, a\}
- **grilo**: \{cricket, annoyance\}
- **da**: \{on the, of, from\}
- **hearth**: \{lareira\}

Using FST
- each rule can be seen as a transducer
Generalising the rule set (FST)

Rules
- o \{the, a\}
- grilo \{cricket, annoyance\}
- da \{on the, of, from\}
- hearth \{lareira\}

Using FST
- each rule can be seen as a transducer
Generalising the rule set (FST)

Rules
- *o* {the, a}
- *grilo* {cricket, annoyance}
- *da* {on the, of, from}
- *hearth* {lareira}

Using FST
- each rule can be seen as a transducer
Generalising the rule set (FST)

Rules
- o \{the, a\}
- grilo \{cricket, annoyance\}
- da \{on the, of, from\}
- hearth \{lareira\}

Using FST
- each rule can be seen as a transducer
Generalising the rule set (FST)

Rules
- o \{the, a\}
- grilo \{cricket, annoyance\}
- da \{on the, of, from\}
- hearth \{lareira\}

Using FST
- each rule can be seen as a transducer
- the union represents the rule set
Generalising the rule set (FST)

Rules
- \(o\): the, a
- \(grilo\): cricket, annoyance
- \(da\): of, from
- \(hearth\): lareira

Using FST
- each rule can be seen as a transducer
- the union represents the rule set
Generalising the rule set (FST)

Rules
- $o$: \{the, a\}
- $grilo$: \{cricket, annoyance\}
- $da$: \{on the, of, from\}
- $hearth$: \{lareira\}

Using FST
- each rule can be seen as a transducer
- the union represents the rule set
- standard intersection mechanisms do the rest
We can translate a lattice encoding the $WL_d$ permutations.
Phrase permutations’ translation with WL$_d$

We can translate a lattice encoding the WL$_d$ permutations

- a truncated window controls reordering
Phrase permutations’ translation with $WL_d$

We can translate a lattice encoding the $WL_d$ permutations

- a truncated window controls reordering
- there is a number of different segmentations of the input

$O(I^2)$ segments

it is sensible to limit phrases to a maximum length

complexity remains linear with sentence length

exponential with distortion limit
Phrase permutations’ translation with $WLD_d$

We can translate a lattice encoding the $WLD_d$ permutations

- a truncated window controls reordering
- there is a number of different segmentations of the input
  - $O(I^2)$ segments
Phrase permutations’ translation with $WL_d$

We can translate a lattice encoding the $WL_d$ permutations

- a truncated window controls reordering
- there is a number of different segmentations of the input
  - $O(I^2)$ segments
  - it is sensible to limit phrases to a maximum length
Phrase permutations’ translation with $WL_d$

We can translate a lattice encoding the $WL_d$ permutations

- a truncated window controls reordering
- there is a number of different segmentations of the input
  - $O(I^2)$ segments
  - it is sensible to limit phrases to a maximum length
- complexity remains
  - linear with sentence length
  - exponential with distortion limit
Generalising the rule set (ITG)

Simply extend the terminal rules
Generalising the rule set (ITG)

Simply extend the terminal rules

- $X \rightarrow XX$
  - direct order

Examples:

- $X \rightarrow o / the$
- $X \rightarrow grilo / cricket$
- $X \rightarrow da / on the$

The intersection mechanisms do the rest

- $O(I_3)$
  - nodes (phrases are limited in length)
- $O(tI_3)$
  - edges
Generalising the rule set (ITG)

Simply extend the terminal rules

- $X \rightarrow XX$
  direct order
- $X \rightarrow \langle XX \rangle$
  inverted order

Examples
- $X \rightarrow o / the$
- $X \rightarrow grilo / cricket$
- $X \rightarrow da / on the$

The intersection mechanisms do the rest

- $O(I_3)$ nodes (phrases are limited in length)
- $O(tI_3)$ edges

31 / 53
Generalising the rule set (ITG)

Simply extend the terminal rules

- $X \rightarrow XX$
  - direct order
- $X \rightarrow \langle XX \rangle$
  - inverted order
- $X \rightarrow r_i$, where $r_i \in R$
  - bilingual mappings

Examples

- $X \rightarrow o$/the
- $X \rightarrow grilo$/cricket
- $X \rightarrow da$/on the

The intersection mechanisms do the rest

- $O(I_3)$ nodes (phrases are limited in length)
- $O(tI_3)$ edges
Generalising the rule set (ITG)

Simply extend the terminal rules

- $X \rightarrow XX$
  - direct order

- $X \rightarrow \langle XX \rangle$
  - inverted order

- $X \rightarrow r_i$, where $r_i \in R$
  - bilingual mappings

Examples

- $X \rightarrow o/the$
- $X \rightarrow grilo/cricket$
- $X \rightarrow da/on the$
Generalising the rule set (ITG)

Simply extend the terminal rules

- $X \rightarrow XX$
  - direct order
- $X \rightarrow \langle XX \rangle$
  - inverted order
- $X \rightarrow r_i$, where $r_i \in R$
  - bilingual mappings

Examples

- $X \rightarrow o / \text{the}$
- $X \rightarrow \text{grilo} / \text{cricket}$
- $X \rightarrow \text{da} / \text{on the}$

The intersection mechanisms do the rest

- $O(I^3)$ nodes (phrases are limited in length)
- $O(tI^3)$ edges
Recap 3

We have
Recap 3

We have

1. defined different models of translational equivalence
Recap 3

We have

1. defined different models of translational equivalence
   ▶ by translating words or phrases
Recap 3

We have

1. defined different models of translational equivalence
   - by translating words or phrases
   - in arbitrary order
Recap 3

We have

1. defined different models of translational equivalence
   - by translating words or phrases
   - in arbitrary order
   - or according to an ITG
We have

1. defined different models of translational equivalence
   - by translating words or phrases
   - in arbitrary order
   - or according to an ITG

2. efficiently represented the set of translations supported by these models for a given input sentence
Recap 3

We have

1. defined different models of translational equivalence
   ▶ by translating words or phrases
   ▶ in arbitrary order
   ▶ or according to an ITG

2. efficiently represented the set of translations supported by these models for a given input sentence
   ▶ trivially expressed in terms of intersection/composition
Recap 3

We have

1. defined different models of translational equivalence
   ▶ by translating words or phrases
   ▶ in arbitrary order
   ▶ or according to an ITG

2. efficiently represented the set of translations supported by these models for a given input sentence
   ▶ trivially expressed in terms of intersection/composition
   ▶ a logic program can do the same
      (sometimes more convenient, e.g. WLd constraints)
Phrase-based SMT [Koehn et al., 2003]
Phrase-based SMT [Koehn et al., 2003]

- the space of solutions grows linearly with input length and exponentially with the distortion limit
Phrase-based SMT [Koehn et al., 2003]

- the space of solutions grows linearly with input length and exponentially with the distortion limit

ITG [Wu, 1997]
Phrase-based SMT [Koehn et al., 2003]
- the space of solutions grows linearly with input length and exponentially with the distortion limit

ITG [Wu, 1997]
- the space of solutions is cubic in length
Remarks

Phrase-based SMT [Koehn et al., 2003]
- the space of solutions grows linearly with input length and exponentially with the distortion limit

ITG [Wu, 1997]
- the space of solutions is cubic in length
- however less efficiently packed, better motivated constraints on reordering
Remarks (hiero)

Hierarchical phrase-based models [Chiang, 2005]

\[X \rightarrow \text{loja de antiguidades} / \text{old curiosity shop}\]

\[X \rightarrow X_1 \text{de} X_2 / X_2 's X_1\]

\[\text{no purely unlexicalised rules}\]

\[\text{same cubic dependency on input length (as ITGs)}\]

\[\text{Other than monotone translation with glue rules}\]
Remarks (hiero)

Hierarchical phrase-based models [Chiang, 2005]

- more general SCFG rules (typically up to 2 nonterminals)

1Other than monotone translation with glue rules
Hierarchical phrase-based models [Chiang, 2005]

- more general SCFG rules (typically up to 2 nonterminals)
- weakly equivalent to an ITG
  (same set of pairs of strings)

\[1\] Other than monotone translation with glue rules
Hierarchical phrase-based models [Chiang, 2005]

- more general SCFG rules (typically up to 2 nonterminals)
- weakly equivalent to an ITG
  (same set of pairs of strings)
- purely lexicalised rules
  e.g. $X \rightarrow \text{loja de antiguidades/old curiosity shop}$
Hierarchical phrase-based models [Chiang, 2005]

- more general SCFG rules (typically up to 2 nonterminals)
- weakly equivalent to an ITG
  (same set of pairs of strings)
- purely lexicalised rules
  e.g. \( X \rightarrow \text{loja de antiguidades} / \text{old curiosity shop} \)
- as well as lexicalised recursive rules
  e.g. \( X \rightarrow X_1 \text{ de } X_2 / X_2 \ 's \ X_1 \)
Hierarchical phrase-based models [Chiang, 2005]

- more general SCFG rules (typically up to 2 nonterminals)
- weakly equivalent to an ITG
  (same set of pairs of strings)
- purely lexicalised rules
  e.g. \( X \rightarrow \text{loja de antiguidades} / \text{old curiosity shop} \)
- as well as lexicalised recursive rules
  e.g. \( X \rightarrow X_1 \text{ de } X_2 / X_2 \text{'s } X_1 \)
- no purely unlexicalised rules\(^1\)

\(^1\)Other than monotone translation with \textit{glue rules}
Remarks (hiero)

Hierarchical phrase-based models [Chiang, 2005]

- more general SCFG rules (typically up to 2 nonterminals)
- weakly equivalent to an ITG
  (same set of pairs of strings)
- purely lexicalised rules
  e.g. $X \rightarrow \text{loja de antiguidades/old curiosity shop}$
- as well as lexicalised recursive rules
  e.g. $X \rightarrow X_1 \text{ de } X_2 / X_2 \text{'s } X_1$
- no purely unlexicalised rules\(^1\)
- same cubic dependency on input length (as ITGs)

\(^1\) Other than monotone translation with glue rules
What are we missing?

We have characterised the set of solutions “backed” by our transfer model
What are we missing?

We have characterised the set of solutions “backed” by our transfer model

- these solutions are unweighted
What are we missing?

We have characterised the set of solutions “backed” by our transfer model

- these solutions are unweighted
- there is no obvious way to discriminate them
What are we missing?

We have characterised the set of solutions “backed” by our transfer model

▶ these solutions are unweighted
▶ there is no obvious way to discriminate them
▶ we cannot make decisions like that
What are we missing?

We have characterised the set of solutions “backed” by our transfer model

- these solutions are unweighted
- there is no obvious way to discriminate them
- we cannot make decisions like that

We are missing a parameterisation of the model

- the scoring function which will guide the decision making process
Linear models

Let’s call derivation
Linear models

Let’s call **derivation**

- a translation string
Linear models

Let’s call **derivation**

- a translation string
- along with any latent structure assumed by the transfer model e.g. phrase segmentation, alignment

A linear parameterisation of the model is a function

$$f(d) = \sum_{k} \lambda_k H_k(d)$$

where $d$ is the derivation, and $H_k$ is one of $m$ feature functions

It assigns a real-valued score to each and every derivation

Think of it as a surrogate for translation quality at decoding time

[Berger et al., 1996]
[Och and Ney, 2002]
Linear models

Let’s call **derivation**

- a translation string
- along with any latent structure assumed by the transfer model  
  e.g. phrase segmentation, alignment

A linear parameterisation of the model is a function

\[ f(d) = \sum_k \lambda_k H_k(d) \]

where \( d \) is the derivation, and \( H_k \) is one of \( m \) feature functions
Linear models

Let’s call **derivation**

- a translation string
- along with any latent structure assumed by the transfer model e.g. phrase segmentation, alignment

A linear parameterisation of the model is a function

$$f(d) = \sum_{k} \lambda_k H_k(d)$$

where $d$ is the derivation, and $H_k$ is one of $m$ feature functions

It assigns a real-valued score to each and every derivation
Linear models

Let’s call **derivation**

- a translation string
- along with any latent structure assumed by the transfer model
e.g. phrase segmentation, alignment

A linear parameterisation of the model is a function

\[ f(d) = \sum_{k} \lambda_k H_k(d) \]

where \(d\) is the derivation, and \(H_k\) is one of \(m\) feature functions

It assigns a real-valued score to each and every derivation

Think of it as a surrogate for translation quality at decoding time

[Berger et al., 1996]
[Och and Ney, 2002]
Feature functions

Independently capture different aspects of the translation, such as

- adequacy
  - translation probabilities
  - confidence on lexical choices
- fluency
  - LM probabilities
  - confidence on reordering
Independence assumptions

Our transfer model makes independence assumptions

▶ “translation happens by concatenating isolated rules” e.g. flat mappings, hierarchical mappings
Independence assumptions

Our transfer model makes independence assumptions

- “translation happens by concatenating isolated rules” e.g. flat mappings, hierarchical mappings

Certain aspects of translation quality comply with such assumptions

- how likely a certain translation rule is
e.g. relative frequency in a bilingual corpus
Structural independence: scoring rules in isolation

Scoring rules independently

0 → 1

1 → 2

2 → 3


nosso

amigo

comum
Structural independence: scoring rules in isolation

Scoring rules independently

0 \rightarrow 1 \rightarrow 2 \rightarrow 3

our/0.6 -> amigo 
ours/0.4 -> comum
 Structural independence: scoring rules in isolation

Scoring rules independently

\[ \text{our}/0.6 \quad \text{friend}/0.7 \]

\[ \text{ours}/0.4 \quad \text{mate}/0.3 \]

\[ \text{comum} \]
Structural independence: scoring rules in isolation

Scoring rules independently

\begin{itemize}
\item our / 0.6
\item ours / 0.4
\item friend / 0.7
\item mate / 0.3
\item usual / 0.4
\item mutual / 0.1
\item ordinary / 0.2
\item common / 0.3
\end{itemize}
Structural independence: scoring rules in isolation

Scoring rules independently

![Graph showing scoring rules]

- 0 (our/0.6, ours/0.4)
- 1 (friend/0.7)
- 2 (mate/0.3, mutual/0.3)
- 2' (mutual)
- 3 (ordinary/0.2, usual/0.4, mutual/0.1, common/0.3, camarada/0.2, friend/0.8)
Structural independence: scoring rules in isolation

Scoring rules independently

inference runs in time linear with the size of the automaton
Independence assumptions

Our transfer model makes independence assumptions

▶ “translation happens by concatenating isolated rules” e.g. flat mappings, hierarchical mappings

Certain aspects of translation quality comply with such assumptions

▶ how likely a certain translation rule is
e.g. relative frequency in a bilingual corpus
Independence assumptions

Our transfer model makes independence assumptions
▶ “translation happens by concatenating isolated rules” e.g. flat mappings, hierarchical mappings

Certain aspects of translation quality comply with such assumptions
▶ how likely a certain translation rule is
e.g. relative frequency in a bilingual corpus

Certain aspects do not comply with such assumptions
▶ fluency as captured by an $n$-gram LM component
Scoring strings with a 2-gram LM

requires unpacking the representation
Scoring strings with a 2-gram LM

requires unpacking the representation
Scoring strings with a 2-gram LM

requires unpacking the representation
Scoring strings with a 2-gram LM

requires unpacking the representation
Scoring strings with a 2-gram LM

requires unpacking the representation
Scoring strings with a 2-gram LM

requires unpacking the representation
Scoring strings with a 2-gram LM

requires unpacking the representation
Scoring strings with a 2-gram LM

requires unpacking the representation
Scoring strings with a 2-gram LM

requires unpacking the representation
Scoring strings with a 2-gram LM

requires unpacking the representation
Imagine a feature function that requires a complete translation.
Scoring whole sentences

Imagine a feature function that requires a complete translation

- unbounded LM
  e.g. via suffix arrays [Zhang and Vogel, 2006]
- estimated overall translation quality
Scoring whole sentences

Imagine a feature function that requires a complete translation

- unbounded LM
  - e.g. via suffix arrays [Zhang and Vogel, 2006]
- estimated overall translation quality

No factorisation at the phrase (nor $n$-gram) level
Scoring whole sentences

Imagine a feature function that requires a complete translation

- unbounded LM
  - e.g. via suffix arrays [Zhang and Vogel, 2006]
- estimated overall translation quality

No factorisation at the phrase (nor $n$-gram) level

- requires fully unpacking the representation
Scoring whole sentences

Imagine a feature function that requires a complete translation

- unbounded LM
  - e.g. via suffix arrays [Zhang and Vogel, 2006]
- estimated overall translation quality

No factorisation at the phrase (nor $n$-gram) level

- requires fully unpacking the representation
- making dependencies explicit through the graphical structure
Scoring whole sentences: example

Exhaustive enumeration

- number of edges exponential with input length
- intractable
Not all is lost

Most features we can reliably estimate
Not all is lost

Most features we can reliably estimate

- are rarely sensitive to global context
Not all is lost

Most features we can reliably estimate
▶ are rarely sensitive to global context
▶ are quite incremental
Not all is lost

Most features we can reliably estimate
  ▶ are rarely sensitive to global context
  ▶ are quite incremental

\( n \)-gram LMs are good examples
  ▶ there are up to \(|\Delta|^{n-1}\) contexts that must be made explicit
Not all is lost

Most features we can reliably estimate

▶ are rarely sensitive to global context
▶ are quite incremental

$n$-gram LMs are good examples

▶ there are up to $|Δ|^n−1$ contexts that must be made explicit
▶ nodes must group derivations sharing the same context
Not all is lost

Most features we can reliably estimate

- are rarely sensitive to global context
- are quite incremental

$n$-gram LMs are good examples

- there are up to $|\Delta|^{n-1}$ contexts that must be made explicit
- nodes must group derivations sharing the same context
- polynomial, though often prohibitive (impracticable)
Recap 4

1. a characterisation the space of solutions
Recap 4

1. a characterisation the space of solutions
2. a linear parameterisation of the model
Recap 4

1. a characterisation the space of solutions
2. a linear parameterisation of the model
3. impact of parameterisation on packed representations
Recap 4

1. a characterisation the space of solutions
2. a linear parameterisation of the model
3. impact of parameterisation on packed representations

What’s left?
Recap 4

1. a characterisation the space of solutions
2. a linear parameterisation of the model
3. impact of parameterisation on packed representations

What’s left?

- more examples of models and impact on representation
  - distance-based reordering
  - lexicalised models
  - a global feature function
- inference algorithms
Recap 4

1. a characterisation the space of solutions
2. a linear parameterisation of the model
3. impact of parameterisation on packed representations

What’s left?

- more examples of models and impact on representation
  - distance-based reordering
  - lexicalised models
  - a global feature function
- inference algorithms
- techniques to make inference feasible for interesting models
Picking one solution

What do we pick out of the (whole) weighted space of solutions?

- best translation
- “minimum-loss” translation
Best translation

MAP

\[ y^* = \arg\max_y \sum_{y[d]=y} f(d|x) \]
Best translation

MAP

\[ y^* = \arg\max_y \sum_{y[d]=y} f(d|x) \]

- summing alternative derivations of the same string
- NP-complete: related to determinisation [Sima'an, 1996]
Best translation

MAP

\[ y^* = \arg\max_y \sum_{y[d]=y} f(d|x) \]

- summing alternative derivations of the same string

NP-complete: related to determinisation [Sima'an, 1996]

Viterbi (approximation to MAP)

\[ d^* = \arg\max_d f(d|x) \]

- assumes the most likely derivation is enough
Minimum Bayes Risk translation

MBR
Minimum Bayes Risk translation

MBR

- incorporates a loss (or gain) function

\[
p(d|x) = \frac{f(d|x)}{\sum_{d'} f(d'|x)}
\]

- can be estimated by sampling translations
- can be estimated from samples of derivations

have a look at project 14 ;)}
Minimum Bayes Risk translation

MBR

- incorporates a loss (or gain) function

\[ y = \arg\min_{y} \langle \text{loss}(y, y') \rangle_{p(y'|x)} \]
Minimum Bayes Risk translation

MBR

- incorporates a loss (or gain) function

\[ y = \arg\max_y \langle \text{gain}(y, y') \rangle_{p(y'|x)} \]
Minimum Bayes Risk translation

MBR

- incorporates a loss (or gain) function

\[ y = \arg\max_y \langle \text{BLEU}(y, y') \rangle_{p(y'|x)} \]
Minimum Bayes Risk translation

MBR

- incorporates a loss (or gain) function

\[
y = \arg \max_y \langle \text{BLEU}(y, y') \rangle_{p(y'|x)}
\]

- assesses the risk associated with choosing any one translation

▶ can be estimated by sampling translations

▶ can be estimated from samples of derivations

▶ have a look at project 14 ;)}
Minimum Bayes Risk translation

MBR

- incorporates a loss (or gain) function

\[ y = \arg\max_y \sum_{y'} \text{BLEU}(y, y') p(y'|x) \]

- assesses the risk associated with choosing any one translation
- requires the computation of expectations

▶ can be estimated by sampling translations
▶ can be estimated from samples of derivations
▶ have a look at project 14 ;)

48 / 53
Minimum Bayes Risk translation

MBR

- incorporates a loss (or gain) function

\[ y = \arg\max_y \sum_{y'} \text{BLEU}(y, y') p(y' | x) \]

- assesses the risk associated with choosing any one translation
- requires the computation of expectations
- which requires a probability

\[ p(d | x) = \frac{f(d | x)}{\sum_{d'} f(d' | x)} \]
Minimum Bayes Risk translation

MBR

- incorporates a loss (or gain) function

\[
y = \arg\max_y \sum_{y' \sim p(y'|x)} \text{BLEU}(y, y')
\]

- assesses the risk associated with choosing any one translation
- requires the computation of expectations
- which requires a probability

\[
p(d|x) = \frac{f(d|x)}{\sum_{d'} f(d'|x)}
\]

- can be estimated by sampling translations
Minimum Bayes Risk translation

MBR

- incorporates a loss (or gain) function

\[ y = \arg\max_y \sum_y \sum_{y'} \text{BLEU}(y, y[y']) \]

- assesses the risk associated with choosing any one translation
- requires the computation of expectations
- which requires a probability

\[ p(d|x) = \frac{f(d|x)}{\sum_{d'} f(d'|x)} \]

- can be estimated by sampling translations
- can be estimated from samples of derivations
Minimum Bayes Risk translation

MBR

- incorporates a loss (or gain) function

\[ y = \arg\max_y \sum_{y'} \sum_{d' \sim p(d'|x)} \text{BLEU}(y, y[d']) \]

- assesses the risk associated with choosing any one translation
- requires the computation of expectations
- which requires a probability

\[ p(d|x) = \frac{f(d|x)}{\sum_{d'} f(d'|x)} \]

- can be estimated by sampling translations
- can be estimated from samples of derivations
- have a look at project 14 ;}

DP-based Viterbi

Explore a truncated version of the full space
DP-based Viterbi

Explore a truncated version of the full space

- only a budgeted set of outgoing edges form each node
DP-based Viterbi

Explore a truncated version of the full space

- only a budgeted set of outgoing edges form each node
  - beam search: exhaustively enumerates outgoing edges, ranks them, prunes all but $k$-best
DP-based Viterbi

Explore a truncated version of the full space

- only a budgeted set of outgoing edges form each node
  - beam search: exhaustively enumerates outgoing edges, ranks them, prunes all but $k$-best
  - cube pruning: enumerates $k$ edges in near best-first order
DP-based Viterbi

Explore a truncated version of the full space

- only a budgeted set of outgoing edges form each node
  - beam search: exhaustively enumerates outgoing edges, ranks them, prunes all but $k$-best
  - cube pruning: enumerates $k$ edges in near best-first order

In order to compare hypotheses more fairly
DP-based Viterbi

Explore a truncated version of the full space
▶ only a budgeted set of outgoing edges form each node
  ▶ beam search: exhaustively enumerates outgoing edges, ranks them, prunes all but $k$-best
  ▶ cube pruning: enumerates $k$ edges in near best-first order

In order to compare hypotheses more fairly
▶ future cost estimates
DP-based Viterbi

Explore a truncated version of the full space

- only a budgeted set of outgoing edges form each node
  - beam search: exhaustively enumerates outgoing edges, ranks them, prunes all but $k$-best
  - cube pruning: enumerates $k$ edges in near best-first order

In order to compare hypotheses more fairly

- future cost estimates
- heuristic view of outside weights
DP-based Viterbi

Explore a truncated version of the full space

- only a budgeted set of outgoing edges form each node
  - beam search: exhaustively enumerates outgoing edges, ranks them, prunes all but $k$-best
  - cube pruning: enumerates $k$ edges in near best-first order

In order to compare hypotheses more fairly

- future cost estimates
- heuristic view of outside weights
- cheap dynamic program that estimates the best possible way to complete any translation prefix
DP-based Viterbi

Explore a truncated version of the full space

- only a budgeted set of outgoing edges form each node
  - beam search: exhaustively enumerates outgoing edges, ranks them, prunes all but $k$-best
  - cube pruning: enumerates $k$ edges in near best-first order

In order to compare hypotheses more fairly

- future cost estimates
- heuristic view of outside weights
- cheap dynamic program that estimates the best possible way to complete any translation prefix

[Koehn et al., 2003]
[Chiang, 2007]
DP-based MBR

Uses derivations in an $n$-best list as samples
DP-based MBR

Uses derivations in an $n$-best list as samples

- arguably poor proxy to samples
DP-based MBR

Uses derivations in an $n$-best list as samples

- arguably poor proxy to samples
- arbitrarily biased (due to pruning)
DP-based MBR

Uses derivations in an \( n \)-best list as samples

▶ arguably poor proxy to samples
▶ arbitrarily biased (due to pruning)
▶ centred around the Viterbi solution by design (due to beam search)
DP-based MBR

Uses derivations in an $n$-best list as samples

- arguably poor proxy to samples
- arbitrarily biased (due to pruning)
- centred around the Viterbi solution by design (due to beam search)

[Kumar and Byrne, 2004]
[Tromble et al., 2008]
Sampling

Gibbs sampling
Sampling

Gibbs sampling

1. start with a draft translation
Sampling

Gibbs sampling

1. start with a draft translation
2. resample from posterior (not all simultaneously):
   segmentation, phrase order, phrase selection

Adaptive rejection sampling

1. design a simpler upperbound (e.g. unigram LM)
2. sample from it
3. assess or reject at the complex distribution (e.g. 5-gram LM)
4. rejected samples motivate refinements of the upperbound
5. repeat 2-3 until acceptance rate is reasonable (e.g. 5-10%)

Importance sampling
Sampling

Gibbs sampling

1. start with a draft translation
2. resample from posterior (not all simultaneously):
   segmentation, phrase order, phrase selection
3. repeat 2
Sampling

Gibbs sampling

1. start with a draft translation
2. resample from posterior (not all simultaneously):
   segmentation, phrase order, phrase selection
3. repeat 2

Adaptive rejection sampling
Sampling

Gibbs sampling
1. start with a draft translation
2. resample from posterior (not all simultaneously):
   segmentation, phrase order, phrase selection
3. repeat 2

Adaptive rejection sampling
1. design a simpler upperbound (e.g. unigram LM)
Sampling

Gibbs sampling
1. start with a draft translation
2. resample from posterior (not all simultaneously):
   segmentation, phrase order, phrase selection
3. repeat 2

Adaptive rejection sampling
1. design a simpler upperbound (e.g. unigram LM)
2. sample from it
Sampling

Gibbs sampling
1. start with a draft translation
2. resample from posterior (not all simultaneously):
   segmentation, phrase order, phrase selection
3. repeat 2

Adaptive rejection sampling
1. design a simpler upperbound (e.g. unigram LM)
2. sample from it
3. assess or reject at the complex distribution (e.g. 5-gram LM)
Sampling

Gibbs sampling

1. start with a draft translation
2. resample from posterior (not all simultaneously):
   segmentation, phrase order, phrase selection
3. repeat 2

Adaptive rejection sampling

1. design a simpler upperbound (e.g. unigram LM)
2. sample from it
3. assess or reject at the complex distribution (e.g. 5-gram LM)
4. rejected samples motivate refinements of the upperbound
Sampling

Gibbs sampling
1. start with a draft translation
2. resample from posterior (not all simultaneously):
   segmentation, phrase order, phrase selection
3. repeat 2

Adaptive rejection sampling
1. design a simpler upperbound (e.g. unigram LM)
2. sample from it
3. assess or reject at the complex distribution (e.g. 5-gram LM)
4. rejected samples motivate refinements of the upperbound
5. repeat 2-3 until acceptance rate is reasonable (e.g. 5-10%)
Sampling

Gibbs sampling
1. start with a draft translation
2. resample from posterior (not all simultaneously):
   segmentation, phrase order, phrase selection
3. repeat 2

Adaptive rejection sampling
1. design a simpler upperbound (e.g. unigram LM)
2. sample from it
3. assess or reject at the complex distribution (e.g. 5-gram LM)
4. rejected samples motivate refinements of the upperbound
5. repeat 2-3 until acceptance rate is reasonable (e.g. 5-10%)

Importance sampling
Sampling

Gibbs sampling
1. start with a draft translation
2. resample from posterior (not all simultaneously): segmentation, phrase order, phrase selection
3. repeat 2

Adaptive rejection sampling
1. design a simpler upperbound (e.g. unigram LM)
2. sample from it
3. assess or reject at the complex distribution (e.g. 5-gram LM)
4. rejected samples motivate refinements of the upperbound
5. repeat 2-3 until acceptance rate is reasonable (e.g. 5-10%)

Importance sampling
▶ you will hear from us (project 14) ;)

280x97 51 / 53
Sampling

Disadvantages

▶ hard to do it without introducing bias
▶ might require large number of samples

Advantages

1. broad view of distribution
2. potential to incorporate arbitrarily complex features (at the sentence level at least)
3. sometimes unbiased
4. ideal for MBR and tuning
5. typically stupid simple to parallelise
Sampling

Disadvantages

▶ hard to do it without introducing bias
Sampling

Disadvantages

- hard to do it without introducing bias
- might require large number of samples
Sampling

Disadvantages
- hard to do it without introducing bias
- might require large number of samples

Advantages
Sampling

Disadvantages
▶ hard to do it without introducing bias
▶ might require large number of samples

Advantages
1. broad view of distribution
Sampling

Disadvantages
- hard to do it without introducing bias
- might require large number of samples

Advantages
1. broad view of distribution
2. potential to incorporate arbitrarily complex features
   (at the sentence level at least)
Sampling

Disadvantages

▶ hard to do it without introducing bias
▶ might require large number of samples

Advantages

1. broad view of distribution
2. potential to incorporate arbitrarily complex features (at the sentence level at least)
3. sometimes unbiased
Sampling

Disadvantages

▶ hard to do it without introducing bias
▶ might require large number of samples

Advantages

1. broad view of distribution
2. potential to incorporate arbitrarily complex features
   (at the sentence level at least)
3. sometimes unbiased
4. ideal for MBR and tuning
Sampling

Disadvantages

- hard to do it without introducing bias
- might require large number of samples

Advantages

1. broad view of distribution
2. potential to incorporate arbitrarily complex features (at the sentence level at least)
3. sometimes unbiased
4. ideal for MBR and tuning
5. typically stupid simple to parallelise
Thanks!

Questions?


References II


