Decoding

Maschinelle Übersetzung

Samuel Läubli

Institut für Computerlinguistik
Universität Zürich

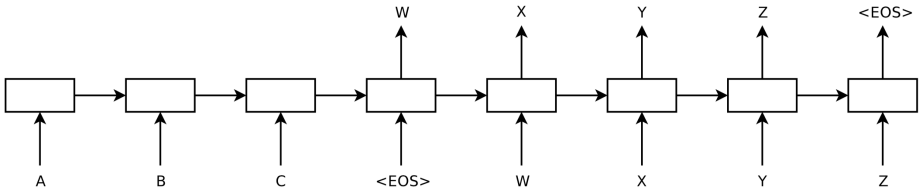
15. Mai 2018

Universität Zürich[UZH]

Themen

1. Einleitung
2. Random Sampling
3. Greedy Search (1-best)
4. Beam Search (k-best)
5. Längennormalisierung
6. Anwendungsspezifisches Decoding
7. Fragen

Wiederholung: Encoder-Decoder-Modell



Quelle: Sutskever et al. (2014)

Wiederholung: Encoder-Decoder-Modell

$$
\begin{aligned}
m_t^{(f)} &= M_{\cdot, f_t}^{(f)} \\
h_t^{(f)} &= \begin{cases} \text{RNN}^{(f)}(m_t^{(f)}, h_{t-1}^{(f)}) & t \geq 1, \\ 0 & \text{otherwise.} \end{cases} \\
m_t^{(e)} &= M_{\cdot, e_{t-1}}^{(e)} \\
h_t^{(e)} &= \begin{cases} \text{RNN}^{(e)}(m_t^{(e)}, h_{t-1}^{(e)}) & t \geq 1, \\ h_{|F|}^{(f)} & \text{otherwise.} \end{cases} \\
p_t^{(e)} &= \text{softmax}(W_{hs} h_t^{(e)} + b_s)
\end{aligned}
$$

Quelle: Neubig (2017, S. 39)

Decoding

Wir bauen einen Satz in der Zielsprache schrittweise – Wort für Wort – auf.

Problem 1: Satzlänge

- Wir bauen einen Satz in der Zielsprache schrittweise auf.
- Nur: Wann hören wir auf?
- Valide Sätze können theoretisch unendlich lang sein.

Problem 1: Satzlänge

- Wir bauen einen Satz in der Zielsprache schrittweise auf.
- Nur: Wann hören wir auf?
- Valide Sätze können theoretisch unendlich lang sein.

$\rightarrow$ Wir generieren Wörter, bis wir das `EOS`-Symbol erhalten.

Problem 1: Satzlänge

- Wir bauen einen Satz in der Zielsprache schrittweise auf.
- Nur: Wann hören wir auf?
- Valide Sätze können theoretisch unendlich lang sein.

$\rightarrow$ Wir generieren Wörter, bis wir das `EOS`-Symbol erhalten.

$\rightarrow$ Abbruchbedingung: Wir generieren höchstens `max_len` Wörter.

Problem 2: Grösse des Suchraums

- Vokabulargrösse $|V|$
- Maximale Satzlänge `max_len`
- Wir können mit unserem Modell $|V|^{max\_len}$ Sätze generieren.

Grösse des Suchraums: Beispiel

- $V = \{$ `EOS`, can, everything, except, I, resist, temptation $\}$
- $|V| = 7$
- `max_len` $= 15$
- Grösse des Suchraums:

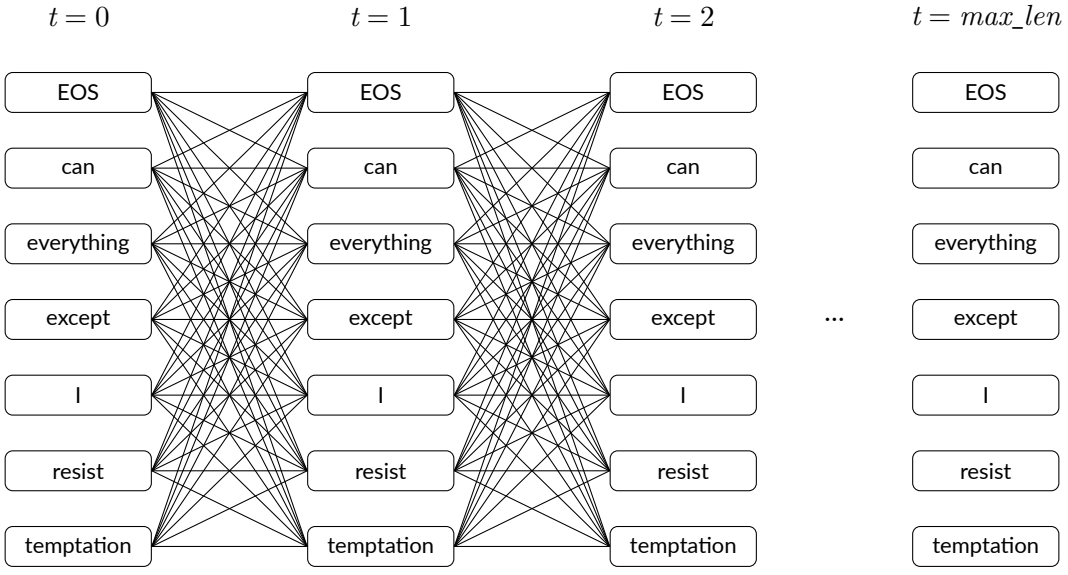
Grösse des Suchraums: Beispiel

- $V = \{$ `EOS`, can, everything, except, I, resist, temptation $\}$
- $|V| = 7$
- `max_len` $= 15$
- Grösse des Suchraums: $7^{15} = 4,747,561,509,943$ Sätze

Grösse des Suchraums: Beispiel

- $V = \{$ `EOS`, can, everything, except, I, resist, temptation $\}$
- $|V| = 7$
- `max_len` $= 15$
- Grösse des Suchraums: $7^{15} = 4,747,561,509,943$ Sätze

$\rightarrow$ Welches ist der beste Satz?


$t = 0$
$t = 1$
$t = 2$
$t = max\_len$
EOS
can
everything
except
I
resist
temptation
EOS
can
everything
except
I
resist
temptation
EOS
can
everything
except
I
resist
temptation
...
EOS
can
everything
except
I
resist
temptation

Decoding: Strategie

- Wir wollen möglichst gute (in NMT: wahrscheinliche) Sätze auswählen.
- Durch die Anwendung der softmax-Funktion erhalten wir für jede Stelle t eine Wahrscheinlichkeitsverteilung über alle Wörter (oder Subwörter) unseres Vokabulars.
- Berechnung der Wahrscheinlichkeit jedes möglichen Satzes ist zu teuer.

$\rightarrow$ Wir brauchen eine Decoding-Strategie, um unwahrscheinliche Sätze möglichst früh zu verwerfen.

Themen

1. Einleitung
2. Random Sampling
3. Greedy Search (1-best)
4. Beam Search (k-best)
5. Längennormalisierung
6. Anwendungsspezifisches Decoding
7. Fragen

Themen

1. Einleitung
2. Random Sampling
3. Greedy Search (1-best)
4. Beam Search (k-best)
5. Längennormalisierung
6. Anwendungsspezifisches Decoding
7. Fragen

Random Sampling[1]

- Wir verfolgen einen einzigen Pfad im Suchraum.
- An jeder Stelle t wählen wir ein Wort e_t.
- Die Wahrscheinlichkeit, dass wir das Wort $e_t \in V$ auswählen, ergibt sich aus der Softmax-Verteilung $P(e_t \mid \hat{e}_1^{t-1})$.

[1] auch «ancestral sampling» oder «forward sampling»

Random Sampling: Beispiel

- $V = \{$ `EOS`, can, everything, except, I, resist, temptation $\}$
- Eingabe: ich kann allem widerstehen, ausser der Versuchung

$t = 0$ $t = 1$ $t = 2$ $t = |E|$

EOS	0.01
can	0.23
everything	0.13
except	0.07
I	0.41
resist	0.09
temptation	0.06

$t = 0$ $t = 1$ $t = 2$ $t = |E|$

EOS	0.01
can	0.23
everything	0.13
except	0.07
I	0.41
resist	0.09
temptation	0.06

$t = 0$ $t = 1$ $t = 2$ $t = |E|$

everything

0.13

$t = 0$ $t = 1$ $t = 2$ $t = |E|$

everything →

EOS	0.04
can	0.28
everything	0.02
except	0.28
I	0.31
resist	0.03
temptation	0.04

0.13

$t = 0$ $t = 1$ $t = 2$ $t = |E|$

everything →

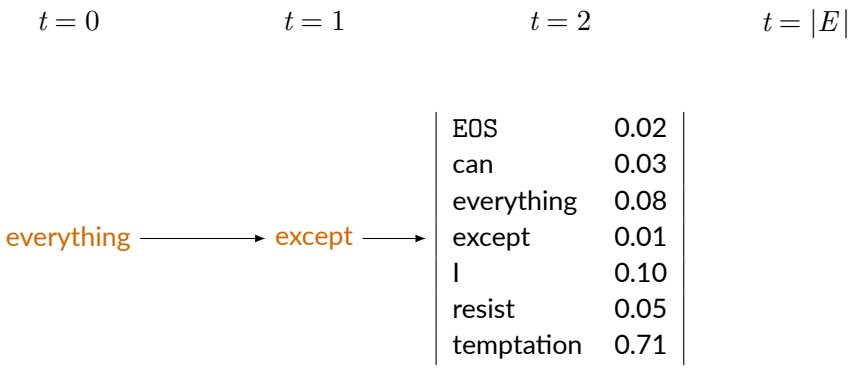
EOS	0.04
can	0.28
everything	0.02
except	0.28
I	0.31
resist	0.03
temptation	0.04

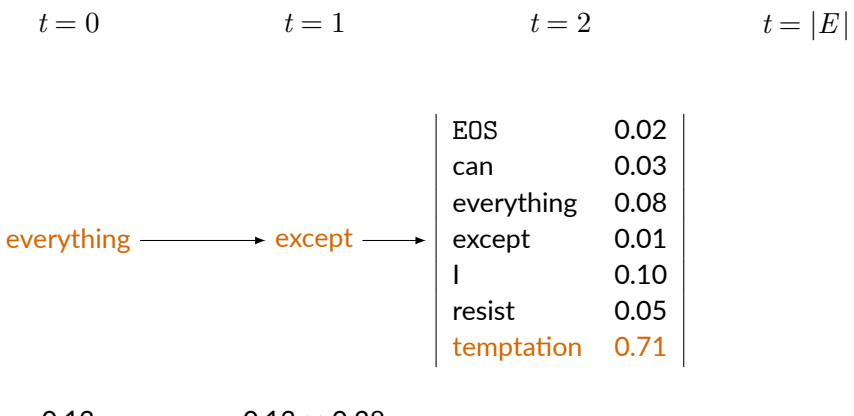
0.13

$t = 0$ $t = 1$ $t = 2$ $t = |E|$

everything ⟶ except

0.13 0.13×0.28


$t = 0$
$t = 1$
$t = 2$
$t = |E|$
everything
except
EOS 0.02
can 0.03
everything 0.08
except 0.01
I 0.10
resist 0.05
temptation 0.71
0.13
0.13 × 0.28


$t = 0$
$t = 1$
$t = 2$
$t = |E|$
everything
except
EOS 0.02
can 0.03
everything 0.08
except 0.01
I 0.10
resist 0.05
temptation 0.71
0.13
0.13 × 0.28

$t = 0$ $t = 1$ $t = 2$ $t = |E|$

everything → except → temptation

0.13 0.13×0.28 $0.13 \times 0.28 \times 0.71$

$t = 0$	$t = 1$	$t = 2$	$t = \lvert E \rvert$

everything ⟶ except ⟶ temptation ⇢ EOS

0.13	0.13×0.28	$0.13 \times 0.28 \times 0.71$	$\prod_{i}^{\lvert E \rvert} P(e_i)$

Random Sampling: Beispiel

- $V = \{$ `EOS`, can, everything, except, I, resist, temptation $\}$
- Eingabe: ich kann allem widerstehen, ausser der Versuchung
- Mögliche Ausgaben (bei mehreren Durchläufen):
  - everything except temptation I can resist
  - I can resist everything except temptation
  - except temptation I can resist everything
  - can I resist everything except temptation
  - I resist can everything except temptation
  - resist

Random Sampling: Beispiel

- $V = \{$ `EOS`, can, everything, except, I, resist, temptation $\}$
- Eingabe: ich kann allem widerstehen, ausser der Versuchung
- Mögliche Ausgaben (bei mehreren Durchläufen):
  - everything except temptation I can resist
  - I can resist everything except temptation
  - except temptation I can resist everything
  - can I resist everything except temptation
  - I resist can everything except temptation $\rightarrow$ wieso?
  - resist

Random Sampling: Beispiel

- $V = \{$ `EOS`, can, everything, except, I, resist, temptation $\}$
- Eingabe: ich kann allem widerstehen, ausser der Versuchung
- Mögliche Ausgaben (bei mehreren Durchläufen):
  - everything except temptation I can resist
  - I can resist everything except temptation
  - except temptation I can resist everything
  - can I resist everything except temptation
  - I resist can everything except temptation $\rightarrow$ wieso?
  - resist $\rightarrow$ wieso?

Random Sampling: Anwendungen

Random Sampling wird verwendet, wenn Variation in der Ausgabe erwünscht ist.

- Dialogsysteme
- Textgenerierung mit Sprachmodellen

Random Sampling: Anwendungen

Random Sampling wird verwendet, wenn Variation in der Ausgabe erwünscht ist.

- Dialogsysteme
- Textgenerierung mit Sprachmodellen

In Romanesco (`sample.py`):

```
for _ in range(length):
    # ...
    next_symbol_probs = softmax(next_symbol_logits)
    # ...
    sampled_symbol = np.random.choice(range(vocab.size), p=next_symbol_probs)
    sampled_sequence.append(sampled_symbol)
```

Themen

1. Einleitung
2. Random Sampling
3. Greedy Search (1-best)
4. Beam Search (k-best)
5. Längennormalisierung
6. Anwendungsspezifisches Decoding
7. Fragen

Greedy Search

Für die maschinelle Übersetzung sind wir i.d.R. an der besten, nicht einer (mehr oder weniger) zufälligen Lösung interessiert.

- Wir verfolgen einen einzigen Pfad im Suchraum.
- An jeder Stelle t wählen wir das Wort e_t mit der höchsten Wahrscheinlichkeit gemäss Softmax-Verteilung $P(e_t \mid \hat{e}_1^{t-1})$ aus.

Greey Search: Beispiel

- $V = \{$ `EOS`, can, everything, except, I, resist, temptation $\}$
- Eingabe: ich kann allem widerstehen, ausser der Versuchung

$t = 0$ $t = 1$ $t = 2$ $t = |E|$

EOS	0.01
can	0.23
everything	0.13
except	0.07
I	0.41
resist	0.09
temptation	0.06

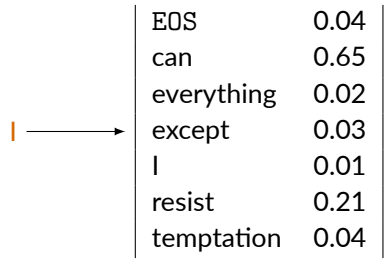
$t = 0$ $t = 1$ $t = 2$ $t = |E|$

EOS	0.01
can	0.23
everything	0.13
except	0.07
I	0.41
resist	0.09
temptation	0.06

$t = 0$ $t = 1$ $t = 2$ $t = |E|$

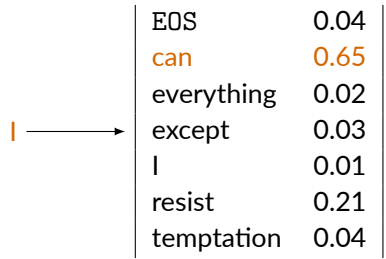
0.41

$t = 0$ $t = 1$ $t = 2$ $t = |E|$



0.41

$t = 0$ $t = 1$ $t = 2$ $t = |E|$



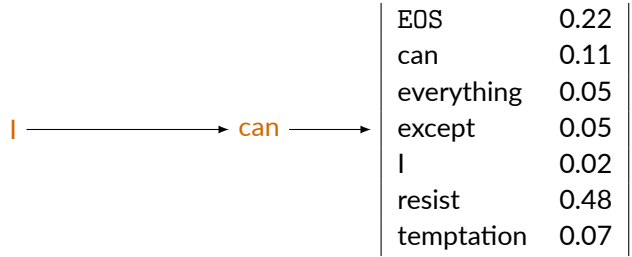
0.41

$t = 0$ $t = 1$ $t = 2$ $t = |E|$

I ⟶ can

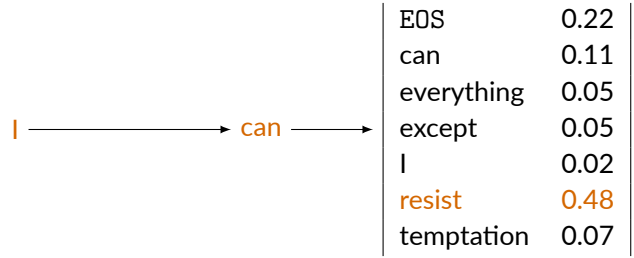
0.41 0.41×0.65

$t = 0$ $t = 1$ $t = 2$ $t = |E|$



0.41 0.41×0.65

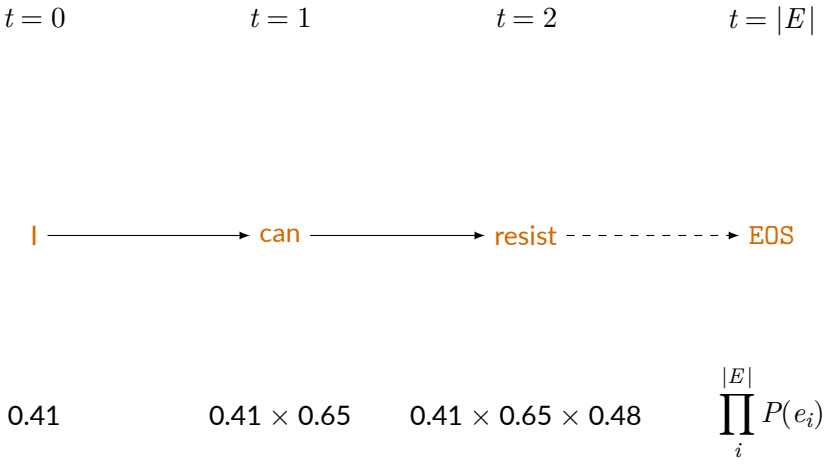
$t = 0$ $t = 1$ $t = 2$ $t = |E|$



0.41 0.41×0.65

$t = 0$ $t = 1$ $t = 2$ $t = |E|$

0.41 0.41×0.65 $0.41 \times 0.65 \times 0.48$


$t = 0$
$t = 1$
$t = 2$
$t = |E|$
I
can
resist
EOS
0.41
0.41 × 0.65
0.41 × 0.65 × 0.48
$\prod_{i}^{|E|} P(e_i)$

Greey Search: Beispiel

- $V = \{$ `EOS`, can, everything, except, I, resist, temptation $\}$
- Eingabe: ich kann allem widerstehen, ausser der Versuchung
- Ausgabe: I can resist everything except temptation
- Immer dieselbe Ausgabe bei mehreren Durchläufen (mit gleichem Modell)

Greedy Search: Problem

Die Greedy-Search-Strategie garantiert nicht, dass wir die Übersetzung mit der höchsten Wahrscheinlichkeit finden.

- Haben wir einmal ein unpassendes Wort gewählt, können wir uns später (= weiter links im Satz) nicht mehr davon «erholen».
- Beispiel:
  - $P(\text{die}) > P(\text{Morgenstund})$ am Satzanfang
  - Aber:
    $P(\text{die Morgenstund hat Gold im Mund}) < P(\text{Morgenstund hat Gold im Mund})$

Themen

1. Einleitung
2. Random Sampling
3. Greedy Search (1-best)
4. Beam Search (k-best)
5. Längennormalisierung
6. Anwendungsspezifisches Decoding
7. Fragen

Beam Search

Um uns von lokalen Fehlentscheidungen «erholen» zu können, verfolgen wir mehrere Lösungen gleichzeitig.

- Wir verfolgen k Pfade im Suchraum.
- In jedem Schritt erweitern wir die k besten Lösungen aus dem vorhergehenden Schritt.
- Sobald wir `EOS` in einer Lösung erreichen, speichern wir sie und fahren mit $k - 1$ Pfaden fort.

Beam Search: Beispiel

- $V = \{$ `EOS`, can, everything, except, I, resist, temptation $\}$
- Eingabe: ich kann allem widerstehen, ausser der Versuchung

$t = 0$ $t = 1$ $t = 2$ $t = 3$

EOS	0.01
can	0.23
everything	0.13
except	0.07
I	0.41
resist	0.09
temptation	0.06

$t=0$	$t=1$	$t=2$	$t=3$

EOS	0.01
can	0.23
everything	0.13
except	0.07
I	0.41
resist	0.09
temptation	0.06

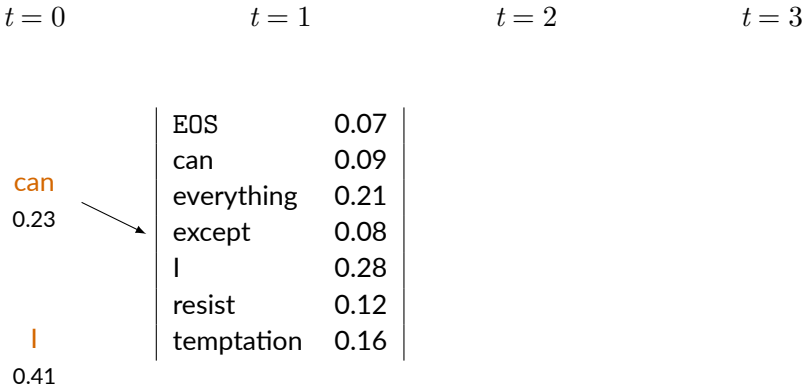
$t = 0$ $t = 1$ $t = 2$ $t = 3$

can

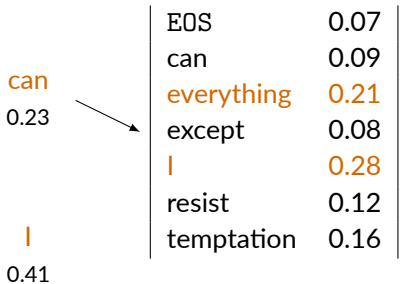
0.23

I

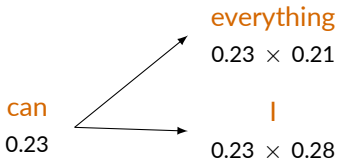
0.41


$t = 0$
$t = 1$
$t = 2$
$t = 3$
can
0.23
I
0.41
EOS 0.07
can 0.09
everything 0.21
except 0.08
I 0.28
resist 0.12
temptation 0.16

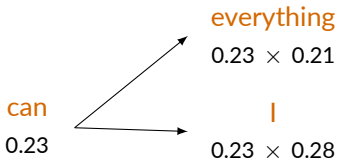
$t = 0$ $t = 1$ $t = 2$ $t = 3$

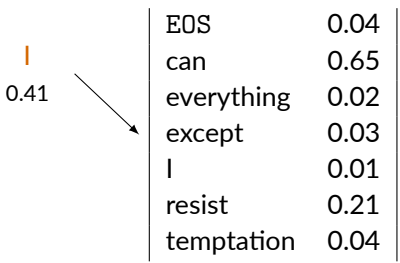

can
0.23
I
0.41
EOS 0.07
can 0.09
everything 0.21
except 0.08
I 0.28
resist 0.12
temptation 0.16

$t = 0$ $t = 1$ $t = 2$ $t = 3$

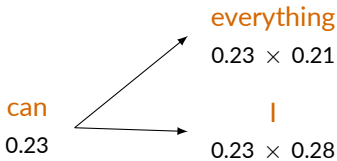


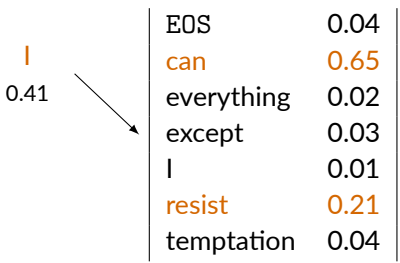
$t = 0$ $t = 1$ $t = 2$ $t = 3$


everything
0.23 × 0.21
can
0.23
I
0.23 × 0.28

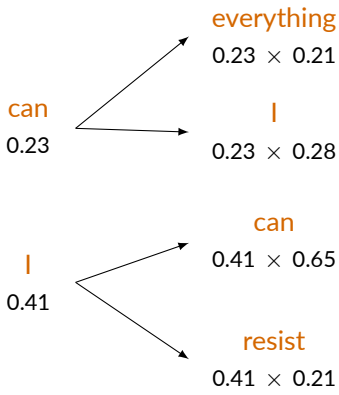

I
0.41
EOS 0.04
can 0.65
everything 0.02
except 0.03
I 0.01
resist 0.21
temptation 0.04

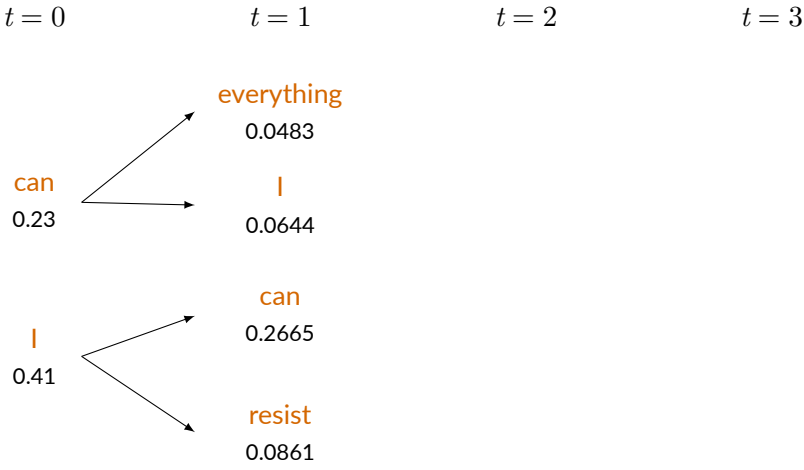
$t = 0$ $t = 1$ $t = 2$ $t = 3$

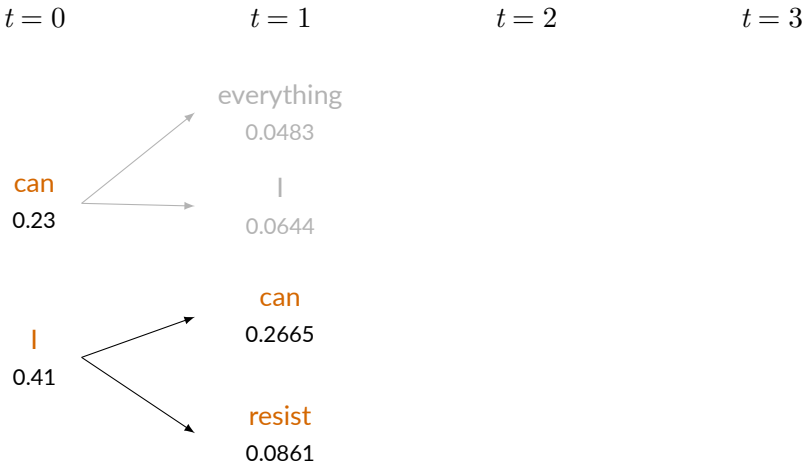

everything
0.23 × 0.21
can
0.23
I
0.23 × 0.28

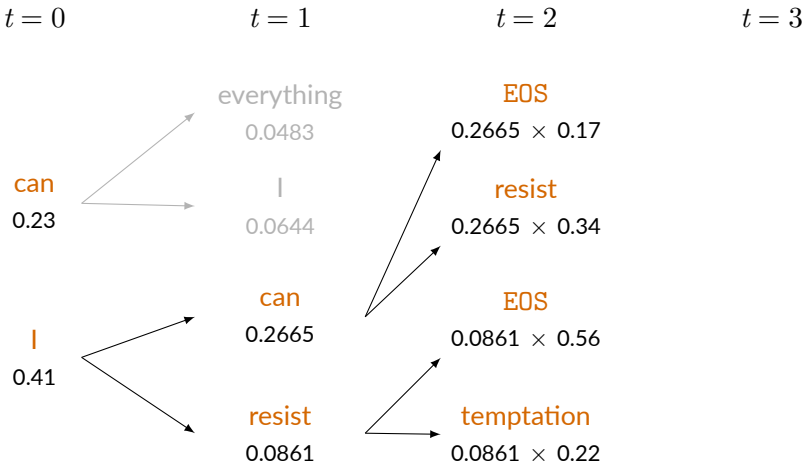

I
0.41
EOS 0.04
can 0.65
everything 0.02
except 0.03
I 0.01
resist 0.21
temptation 0.04

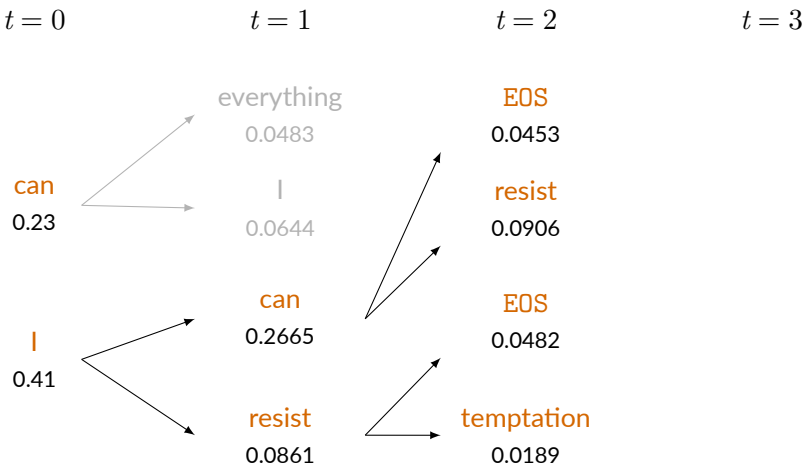
$t = 0$ $t = 1$ $t = 2$ $t = 3$

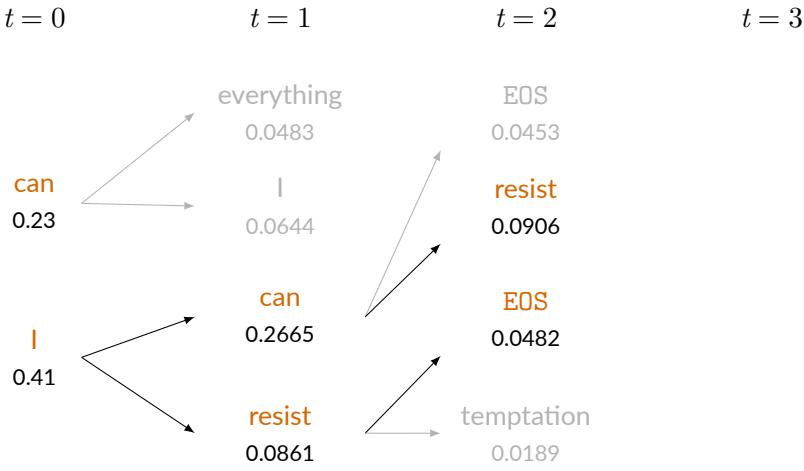

everything
0.23 × 0.21
can
0.23
I
0.23 × 0.28
can
0.41 × 0.65
I
0.41
resist
0.41 × 0.21

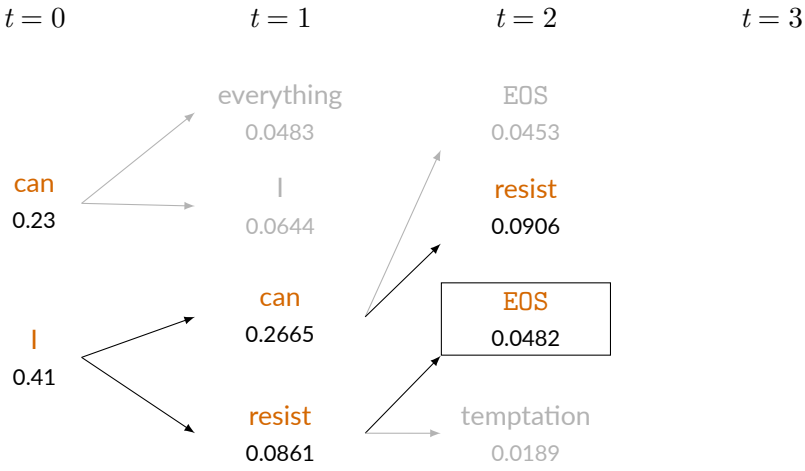

$t = 0$
$t = 1$
$t = 2$
$t = 3$
everything
0.0483
can
0.23
I
0.0644
can
0.2665
I
0.41
resist
0.0861

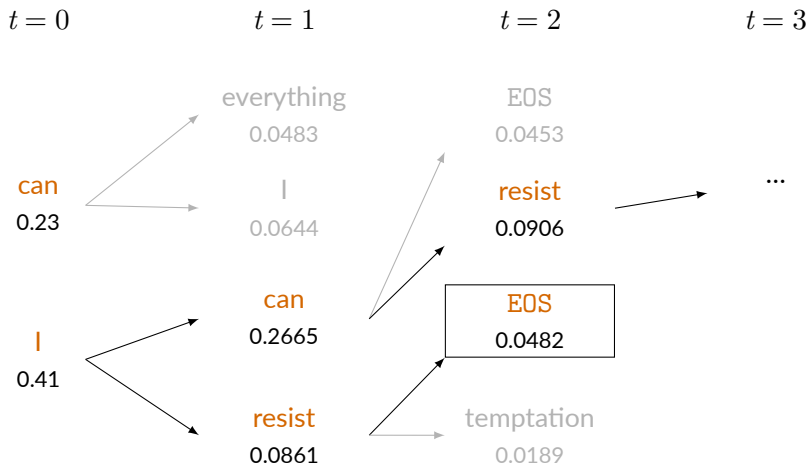

$t = 0$
$t = 1$
$t = 2$
$t = 3$
everything
0.0483
can
0.23
I
0.0644
can
0.2665
I
0.41
resist
0.0861


$t = 0$
$t = 1$
$t = 2$
$t = 3$
can
0.23
I
0.41
everything
0.0483
I
0.0644
can
0.2665
resist
0.0861
EOS
0.2665 × 0.17
resist
0.2665 × 0.34
EOS
0.0861 × 0.56
temptation
0.0861 × 0.22


$t = 0$
$t = 1$
$t = 2$
$t = 3$
everything
0.0483
can
0.23
I
0.0644
EOS
0.0453
resist
0.0906
I
0.41
can
0.2665
EOS
0.0482
resist
0.0861
temptation
0.0189


$t = 0$
$t = 1$
$t = 2$
$t = 3$
everything
0.0483
EOS
0.0453
can
0.23
I
0.0644
resist
0.0906
I
0.41
can
0.2665
EOS
0.0482
resist
0.0861
temptation
0.0189


$t = 0$
$t = 1$
$t = 2$
$t = 3$
everything
0.0483
EOS
0.0453
can
0.23
I
0.0644
resist
0.0906
I
0.41
can
0.2665
EOS
0.0482
resist
0.0861
temptation
0.0189


$t = 0$
$t = 1$
$t = 2$
$t = 3$
can
0.23
I
0.41
everything
0.0483
I
0.0644
can
0.2665
resist
0.0861
EOS
0.0453
resist
0.0906
EOS
0.0482
temptation
0.0189
...

Beam Search: Beispiel

- $V = \{$ `EOS`, can, everything, except, I, resist, temptation $\}$
- Eingabe: ich kann allem widerstehen, ausser der Versuchung
- Ausgabe mit $k = 2$
  - I resist, score = 0.0482
  - I can resist everything except temptation, score = 0.0162

Beam Search: Beispiel

- $V = \{$ EOS, can, everything, except, I, resist, temptation $\}$
- Eingabe: ich kann allem widerstehen, ausser der Versuchung
- Ausgabe mit $k = 2$
  - I resist, score = 0.0482 $\rightarrow$ Wieso besser?
  - I can resist everything except temptation, score = 0.0162

Themen

1. Einleitung
2. Random Sampling
3. Greedy Search (1-best)
4. Beam Search (k-best)
5. Längennormalisierung
6. Anwendungsspezifisches Decoding
7. Fragen

Längennormalisierung

«Without some form of length-normalization regular beam search will favor shorter results over longer ones on average since a negative log-probability is added at each step, yielding lower (more negative) scores for longer sentences.» (Wu et al., 2016)

Längennormalisierung: Division durch Länge

Ohne Längennormalisierung:

I resist, score = $0.0482 \rightarrow -1.3169$ $(\log_{10})$

I can resist everything except temptation, score = $0.0162 \rightarrow -1.7905$ $(\log_{10})$

Log-Score durch Länge (Anzahl Wörter) dividieren:

I resist, score = -1.3169

I can resist everything except temptation, score = -1.7905

Längennormalisierte Scores $\approx$ durchschnittliche Wahrscheinlichkeit pro Wort

Längennormalisierung: Division durch Länge

Ohne Längennormalisierung:

- I resist, score = $0.0482 \rightarrow -1.3169$ $(\log_{10})$
- I can resist everything except temptation, score = $0.0162 \rightarrow -1.7905$ $(\log_{10})$

Log-Score durch Länge (Anzahl Wörter) dividieren:

- I resist, score = $-1.3169/2 = -0.6585$ (0.2195)
- I can resist everything except temptation, score = -1.7905

Längennormalisierte Scores $\approx$ durchschnittliche Wahrscheinlichkeit pro Wort

Längennormalisierung: Division durch Länge

Ohne Längennormalisierung:

- I resist, score = $0.0482 \rightarrow -1.3169$ $(\log_{10})$
- I can resist everything except temptation, score = $0.0162 \rightarrow -1.7905$ $(\log_{10})$

Log-Score durch Länge (Anzahl Wörter) dividieren:

- I resist, score = $-1.3169/2 = -0.6585$ (0.2195)
- I can resist everything except temptation, score = $-1.7905/6 = -0.2984$ (0.5030)

Längennormalisierte Scores $\approx$ durchschnittliche Wahrscheinlichkeit pro Wort

Längennormalisierung: Alpha-Parameter (Wu et al., 2016)

$$\text{score}^* = \frac{\text{score}}{\text{len}(\text{score})^\alpha}$$

- $0 < \alpha < 1$
- $\alpha = 0 \rightarrow$ Keine Längennormalisierung
- $\alpha = 1 \rightarrow$ Maximale Längennormalisierung (entspricht Division durch Länge)
- α-Parameter wird auf Dev-Set optimiert.
- In der Praxis sind Werte zwischen 0.6 und 0.7 ideal (Wu et al., 2016).

Längennormalisierung: Alpha-Parameter (Beispiel)

Ohne Längennormalisierung:

hyp_1: I resist, score = −1.32

hyp_2: I can resist everything except temptation, score = −1.79

Mit Längennormalisierung (Alpha-Parameter):

α	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
hyp_1	-1.32	-1.23	-1.15	-1.07	-1.00	-0.93	-0.87	-0.81	-0.76	-0.71	-0.66
hyp_2	-1.79	-1.50	-1.25	-1.05	-0.87	-0.73	-0.61	-0.51	-0.43	-0.36	-0.30

Längennormalisierung: Übung

Gegeben ist die Ausgabe eines MÜ-Systems mit folgenden Wortwahrscheinlichkeiten:

the	truth	is	rarely	pure	and	never	simple	.	EOS
0.6	0.3	0.5	0.3	0.4	0.6	0.4	0.3	0.8	0.9

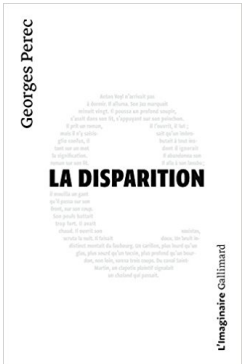
Berechnen Sie den

- Score der Ausgabe
- Log-Score der Ausgabe
- längennormalisierten Log-Score der Ausgabe (Division durch Länge)
- längennormalisierten Log-Score der Ausgabe mit $\alpha = 0.6$

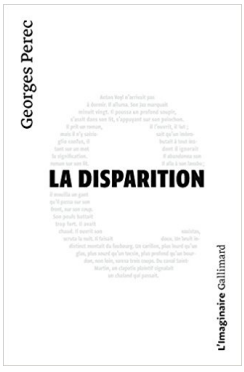
Themen

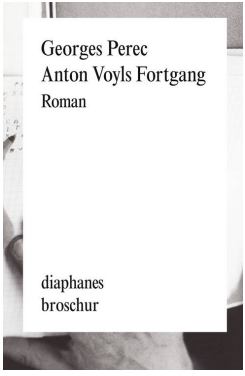
1. Einleitung
2. Random Sampling
3. Greedy Search (1-best)
4. Beam Search (k-best)
5. Längennormalisierung
6. Anwendungsspezifisches Decoding
7. Fragen

Anwendungsbeispiel «La Disparition»



Anwendungsbeispiel «La Disparition»





Anwendungsbeispiel «La Disparation»

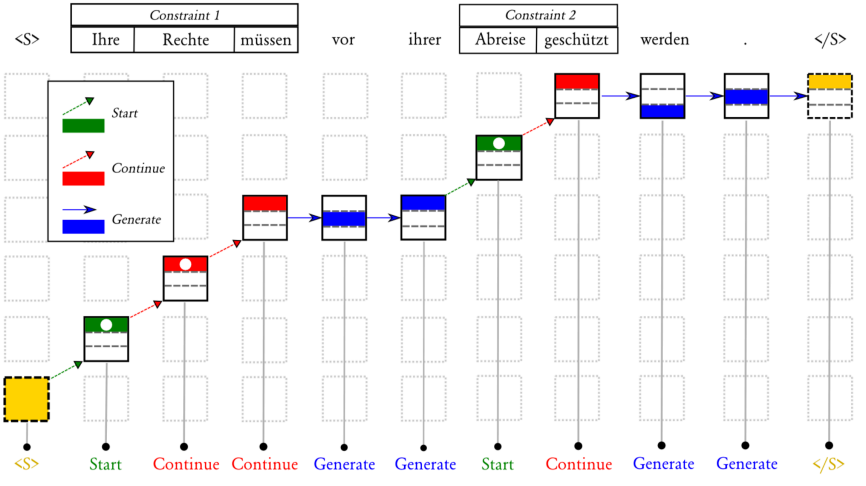
```
for _ in range(length):
  # ...
  next_symbol_probs = softmax(next_symbol_logits)
  # ...
  for word_id, prob in enumerate(next_symbol_probs):
      if not 'e' in vocab.get_word(word_id):
          sampled_sequence.append(sampled_symbol)
          break
```

Lexikalische Constraints (Terminologie)

«In meinem System muss ‹chewing gum› immer als ‹zu kauender Gummi› übersetzt werden.»

Wie forcieren wir Teilübersetzungen (Terminologie, etc.) mit NMT-Systemen?

Grid Beam Search (Hokamp und Liu, 2017)



Input: Rights protection should begin before their departure .

Themen

1. Einleitung
2. Random Sampling
3. Greedy Search (1-best)
4. Beam Search (k-best)
5. Längennormalisierung
6. Anwendungsspezifisches Decoding
7. Fragen

Literatur

Hokamp, Chris und Qun Liu. 2017. Lexically constrained decoding for sequence generation using grid beam search. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada, pages 1535–1546.

Neubig, Graham. 2017. Neural machine translation and sequence-to-sequence models: A tutorial. *arXiv preprint* 1703.01619.

Sutskever, Ilya, Oriol Vinyals, und Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS)*. Montreal, Canada, pages 3104–3112.

Wu, Yonghui, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, und Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint* (arXiv:1609.08144v2).