Diverse decoding methods for NLP

Slides by Farhan Samir and Ganesh Jawahar

Autoregressive conditional neural language models

$$\hat{\mathbf{y}} = \arg\max_{\mathbf{y}} P(\mathbf{y}|\mathbf{x}) = \arg\max_{\mathbf{y}} \prod_{t=1}^{N} P(y_t \mid \overbrace{y_{<t}}^{\text{Prev. output tokens}}, \underbrace{\mathbf{x}}_{\text{Input tokens}})$$

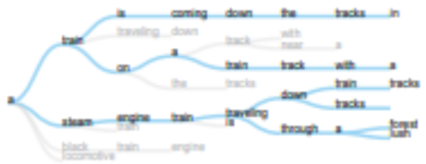
Why do we want diverse text generation?

Ground Truth Captions

Single engine train rolling down the tracks. A locomotive drives along the tracks amongst trees and bushes. An engine is coming down the train track.

A steam locomotive is blowing steam. An old fashion train with steam coming out of its pipe. A black and red train moving down a train track.

Why do we want diverse text generation?



Beam Search

A steam engine train travelling down train tracks.
A steam engine train travelling down tracks.
A steam engine train travelling through a forest.
A steam engine train travelling through a lush green forest.
A steam engine train travelling through a lush green countryside
A train on a train track with a sky background.

Ground Truth Captions

Single engine train rolling down the tracks. A locomotive drives along the tracks amongst trees and bushes. An engine is coming down the train track.
A steam locomotive is blowing steam. An old fashion train with steam coming out of its pipe. A black and red train moving down a train track.

Two simple methods for diverse generation

1. Deterministic: beam search

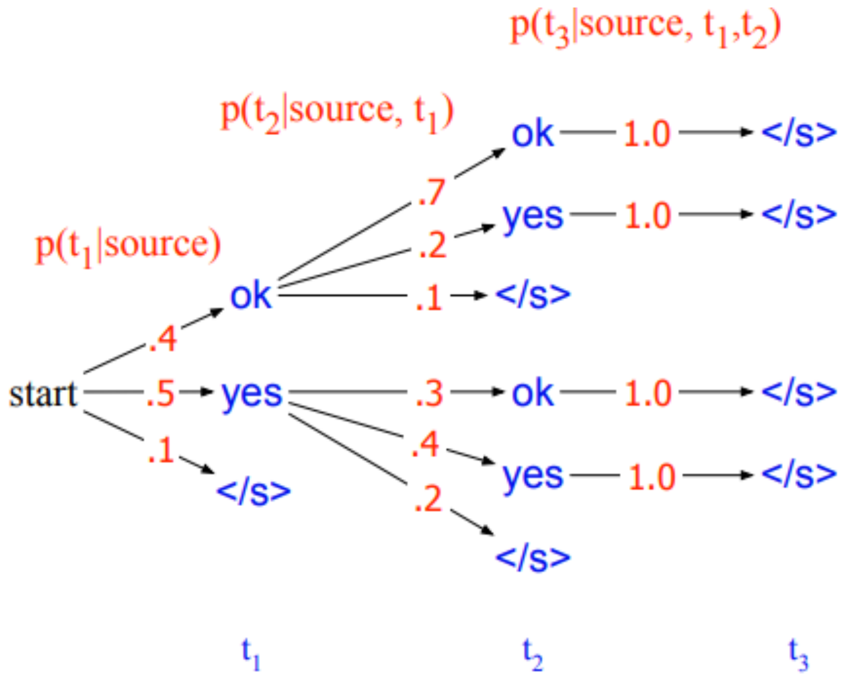
Two simple methods for diverse generation

1. Deterministic: beam search

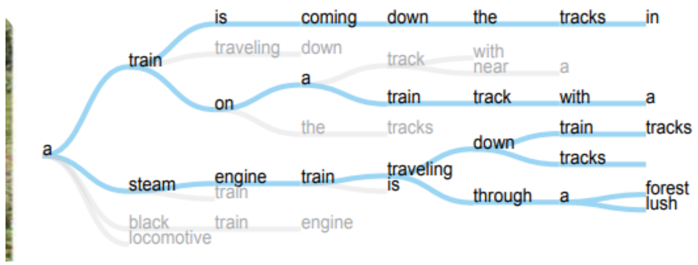
“The **local beam search** algorithm keeps track of **k states rather than just one**. It begins with **k randomly generated states**. At each step, all the **successors of all k states** are generated. If any one state is a goal, the algorithm halts. Otherwise, it selects the k best successors from the complete list and repeats.” - Russell & Norvig (1994)

Two simple methods for diverse generation

1. Deterministic: beam search



Two simple methods for diverse generation



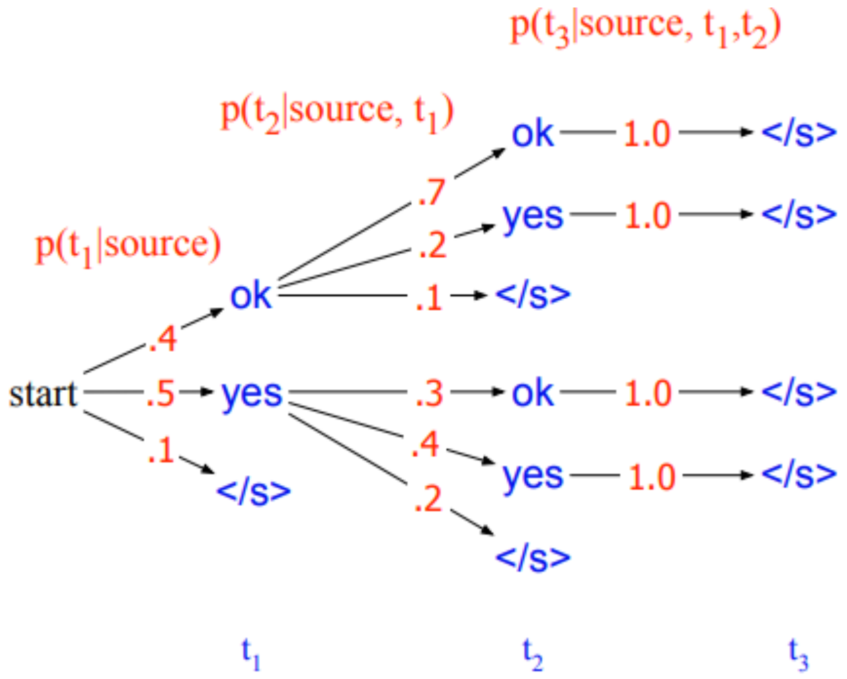
A steam engine train travelling down train tracks.
A steam engine train travelling down tracks.
A steam engine train travelling through a forest.
A steam engine train travelling through a lush green forest.
A steam engine train travelling through a lush green countryside
A train on a train track with a sky background.

Two simple methods for diverse generation

1. Deterministic: beam search
2. Stochastic: ancestral sampling

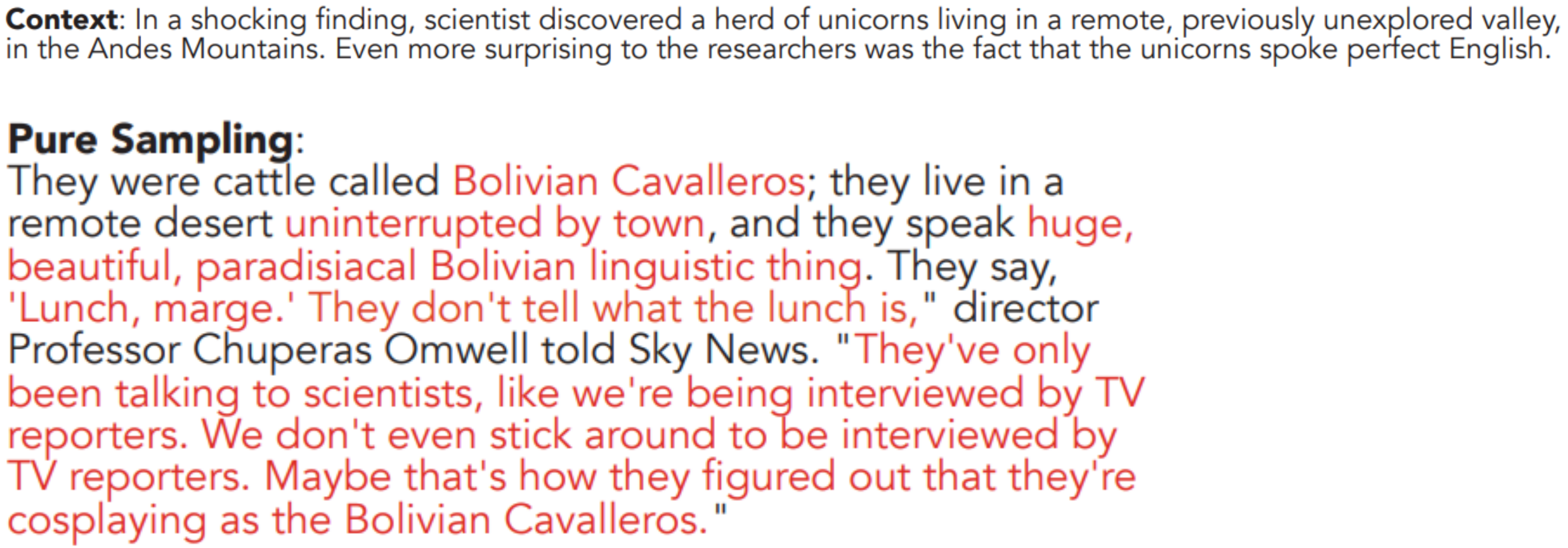
Two simple methods for diverse generation

2. Probabilistic: ancestral sampling



Two simple methods for diverse generation

2. Probabilistic: ancestral sampling



Context: In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

Pure Sampling:
They were cattle called Bolivian Cavalleros; they live in a remote desert uninterrupted by town, and they speak huge, beautiful, paradisiacal Bolivian linguistic thing. They say, 'Lunch, marge.' They don't tell what the lunch is," director Professor Chuperas Omwell told Sky News. "They've only been talking to scientists, like we're being interviewed by TV reporters. We don't even stick around to be interviewed by TV reporters. Maybe that's how they figured out that they're cosplaying as the Bolivian Cavalleros."

Questions?

Search-based diverse generations

1. Delayed Beam Search (Massarelli et al., EMNLP Findings, 2020)
2. Noisy-parallel approximate decoding (Cho, 2016)
3. Clustering post-decoding (Ippolito et al., ACL, 2019)
4. Diverse Decoding Using Signatures (Weir et al., EMNLP, 2020)

Delayed beam search

1. Ancestral sampling for **m steps**
2. Beam search for **remainder**

Delayed beam search

1. Ancestral sampling for **m steps**
2. Beam search for **remainder**

Intuition:

- Induce diversity early in the search tree; refine with beam-search later in the tree

Advantages

1. **Fast**: minimal changes to underlying LM
2. **Extensible**: m could be a function of source sentence length

Disadvantages

1. No theoretical guarantee that ancestral sampling will be stable for m steps

Noisy parallel approximate decoding (NPAD)

$$\mathbf{h}_t = \phi\left(\mathbf{h}_{t-1} + \epsilon_t, \overbrace{\mathbf{E}\left[x_t\right]}^{\text{embedding of time t input}}, f(Y, t)\right),$$

$$\epsilon_t \sim \mathcal{N}(\mathbf{0}, \sigma_t^2 \mathbf{I}).$$

$$\left.\sigma_t = \frac{\sigma_0}{t}\right\rangle \text{decays to 0 over time}$$

Noisy parallel approximate decoding (NPAD)

$$\mathbf{h}_t = \phi\left(\mathbf{h}_{t-1} + \epsilon_t, \overbrace{\mathbf{E}\left[x_t\right]}^{\text{embedding of time t input}}, f(Y, t)\right),$$

$$\epsilon_t \sim \mathcal{N}(\mathbf{0}, \sigma_t^2 \mathbf{I}).$$

$$\left.\sigma_t = \frac{\sigma_0}{t}\right\rangle \text{decays to 0 over time}$$

A major difference between this **sampling-at-the-output** and the proposed **NPAD** is that the NPAD **exploits the hidden state space of a neural network** in which the data manifold is highly linearized. In other words, training a neural network tends to fill up the hidden state space as much as possible with valid data points, and consequently **any point in the neighbourhood of a valid hidden state should map to a plausible point in the output space**. This is contrary to the actual output space, where only a fraction of the output space is plausible

Noisy parallel approximate decoding (NPAD)

$$\mathbf{h}_t = \phi\left(\mathbf{h}_{t-1} + \epsilon_t, \overset{\text{embedding of time t input}}{\frown}\!\!\!\!\!\!\mathbf{E}\left[x_t\right], f(Y,t)\right),$$

$$\epsilon_t \sim \mathcal{N}(\mathbf{0}, \sigma_t^2\mathbf{I}).$$

$$\left.\sigma_t = \frac{\sigma_0}{t}.\right\rangle \text{decays to 0 over time}$$

Advantages

1. **Fast**: minimal changes to underlying LM

Disadvantages

1. Unclear how this would be implemented for non-recurrent LMs (e.g., transformers).

Clustering Post-Decoding

1. Beam-search with very large beam (e.g., **B=100**)
2. **Encode** all outputs (with, e.g., SBERT)
3. Cluster all encoded outputs (with, e.g., k-Means)
4. Select the **highest ranked output** from each cluster

Clustering Post-Decoding

1. Beam-search with very large beam (e.g., **B=100**)
2. **Encode** all outputs (with, e.g., SBERT)
3. Cluster all encoded outputs (with, e.g., k-Means)
4. Select the **highest ranked output** from each cluster

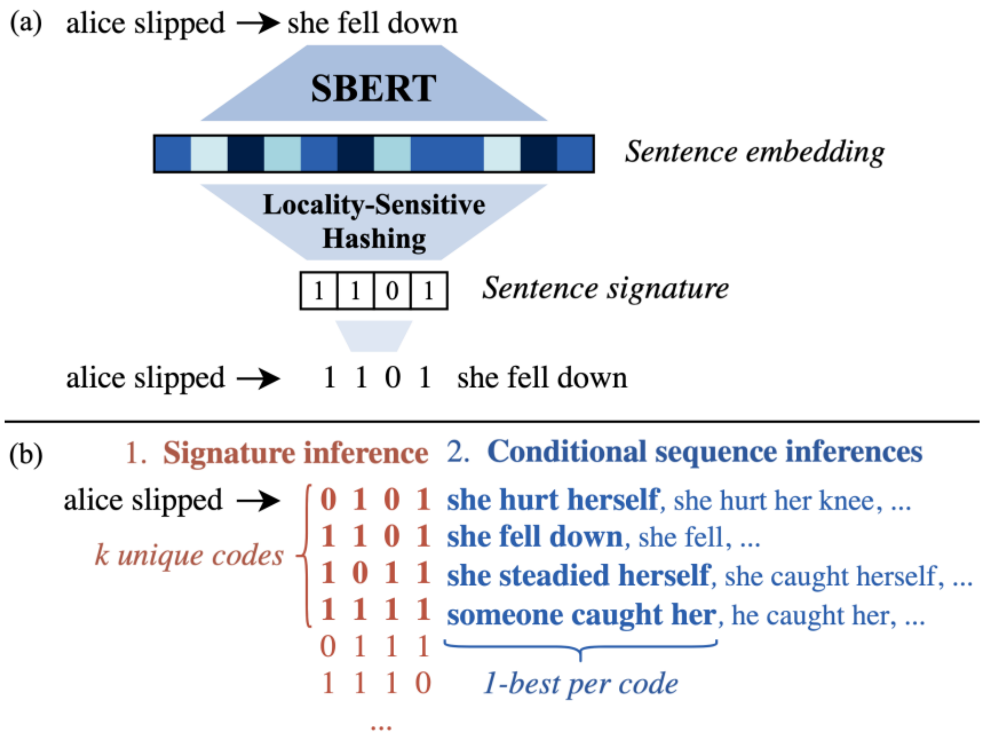
Advantages

1. Works well for preserving **fluency** and **increasing diversity** (Ippolito et al., 2019)

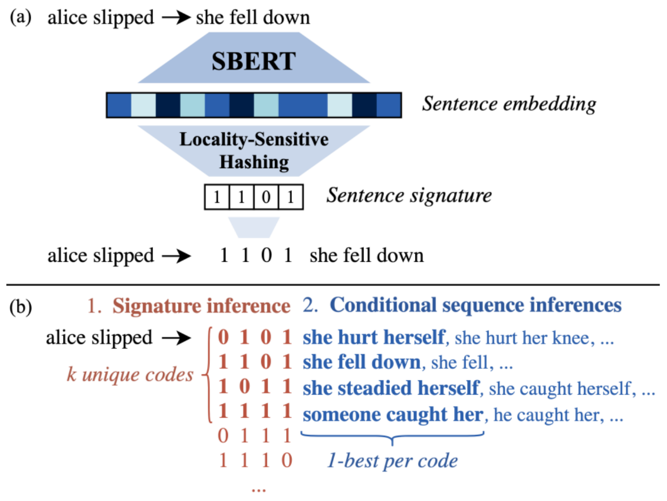
Disadvantages

1. Significant increases in time and space complexity due to:
   a. Encoding
   b. Clustering
   c. Using a larger beam (space complexity and time complexity for beam search increases quadratically in size of beam)

COnstrained Decoding with Semantic Sentence Signatures (Cod3s)


(a) alice slipped → she fell down
SBERT
Sentence embedding
Locality-Sensitive Hashing
1 1 0 1
Sentence signature
alice slipped → 1 1 0 1 she fell down
(b) 1. Signature inference 2. Conditional sequence inferences
alice slipped →
k unique codes
0 1 0 1 she hurt herself, she hurt her knee, ...
1 1 0 1 she fell down, she fell, ...
1 0 1 1 she steadied herself, she caught herself, ...
1 1 1 1 someone caught her, he caught her, ...
0 1 1 1
1 1 1 0
...
1-best per code

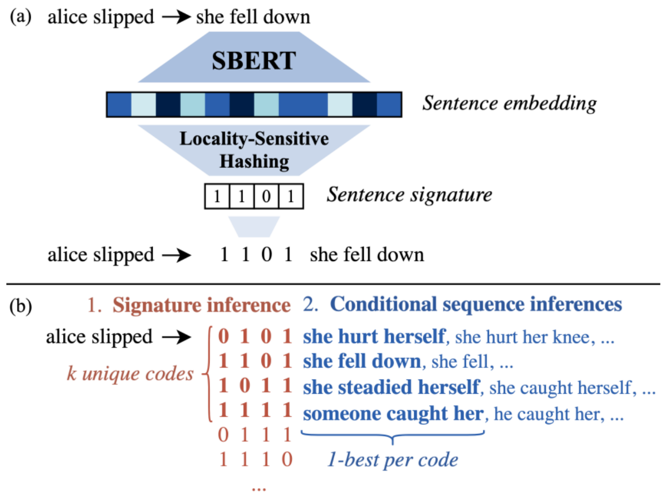
COnstrained Decoding with Semantic Sentence Signatures (Cod3s)


(a) alice slipped → she fell down
SBERT
Sentence embedding
Locality-Sensitive Hashing
1 1 0 1
Sentence signature
alice slipped → 1 1 0 1 she fell down
(b) 1. Signature inference 2. Conditional sequence inferences
alice slipped →
0 1 0 1 she hurt herself, she hurt her knee, ...
1 1 0 1 she fell down, she fell, ...
1 0 1 1 she steadied herself, she caught herself, ...
1 1 1 1 someone caught her, he caught her, ...
0 1 1 1
1 1 1 0
...
k unique codes
1-best per code

Training:

1. Locality sensitive hashing on SBERT(target sentence)-> bit vector

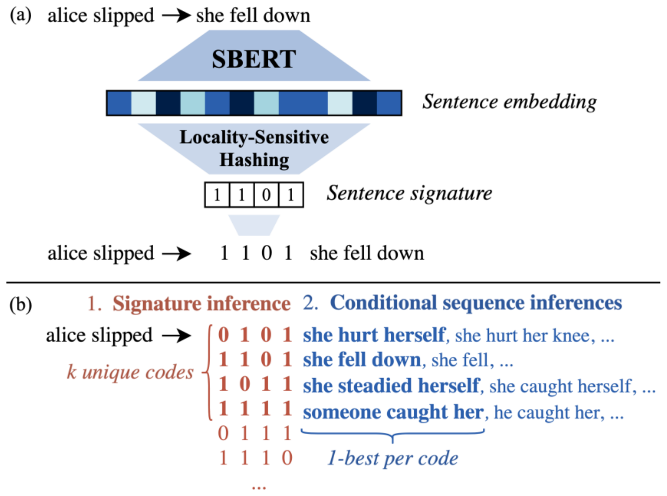
COnstrained Decoding with Semantic Sentence Signatures (Cod3s)


(a) alice slipped → she fell down
SBERT
Sentence embedding
Locality-Sensitive Hashing
1 1 0 1 Sentence signature
alice slipped → 1 1 0 1 she fell down
(b) 1. Signature inference 2. Conditional sequence inferences
alice slipped →
k unique codes
0 1 0 1 she hurt herself, she hurt her knee, ...
1 1 0 1 she fell down, she fell, ...
1 0 1 1 she steadied herself, she caught herself, ...
1 1 1 1 someone caught her, he caught her, ...
0 1 1 1
1 1 1 0
...
1-best per code

Training:

1. Locality sensitive hashing on SBERT(target sentence)-> bit vector
2. Encode source sentence

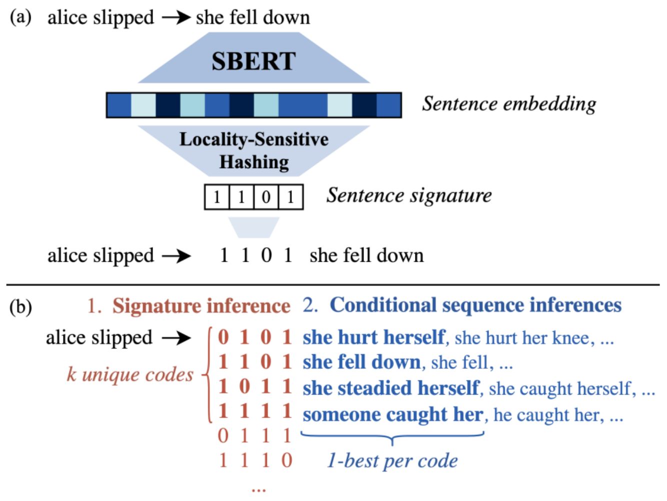
COnstrained Decoding with Semantic Sentence Signatures (Cod3s)


(a) alice slipped → she fell down
SBERT
Sentence embedding
Locality-Sensitive Hashing
1 1 0 1
Sentence signature
alice slipped → 1 1 0 1 she fell down
(b) 1. Signature inference 2. Conditional sequence inferences
alice slipped →
0 1 0 1 she hurt herself, she hurt her knee, ...
1 1 0 1 she fell down, she fell, ...
1 0 1 1 she steadied herself, she caught herself, ...
1 1 1 1 someone caught her, he caught her, ...
0 1 1 1
1 1 1 0
...
k unique codes
1-best per code

Training:

1. Locality sensitive hashing on SBERT(target sentence)-> bit vector
2. Encode source sentence
3. Try to predict [1 1 0 1 she fell down]

COnstrained Decoding with Semantic Sentence Signatures (Cod3s)



(a) alice slipped → she fell down
SBERT
Sentence embedding
Locality-Sensitive Hashing
1 1 0 1
Sentence signature
alice slipped → 1 1 0 1 she fell down
(b) 1. Signature inference 2. Conditional sequence inferences
alice slipped →
k unique codes
0 1 0 1 she hurt herself, she hurt her knee, ...
1 1 0 1 she fell down, she fell, ...
1 0 1 1 she steadied herself, she caught herself, ...
1 1 1 1 someone caught her, he caught her, ...
0 1 1 1
1 1 1 0
1-best per code
...

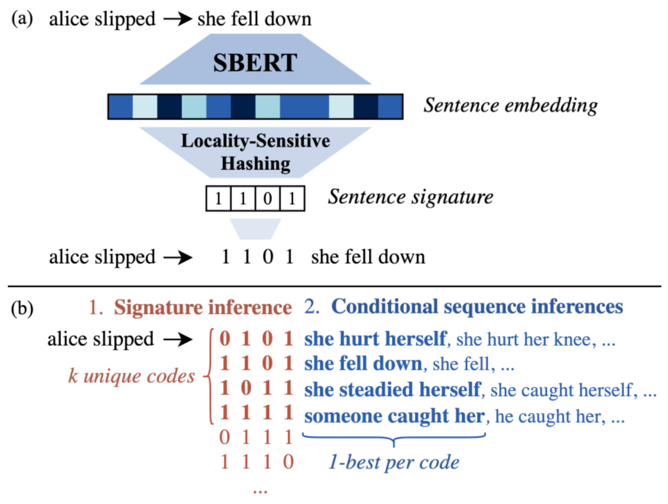
Training:

1. Locality sensitive hashing on SBERT(target sentence)-> bit vector
2. Encode source sentence
3. Try to predict [1 1 0 1 she fell down]

Inference

1. Predict code B codes from source sentence using beam-search
2. Select top k (out of B) codes that have high probability **and** sufficiently high Hamming-distance (greedy algorithm)
3. Generate k predictions conditioning on source and signature using beam-search

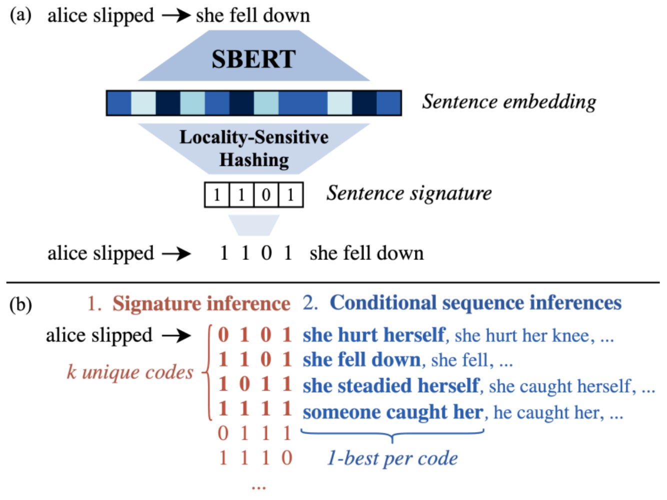
COnstrained Decoding with Semantic Sentence Signatures (Cod3s)


(a) alice slipped → she fell down
SBERT
Sentence embedding
Locality-Sensitive Hashing
1 1 0 1
Sentence signature
alice slipped → 1 1 0 1 she fell down
(b)
1. Signature inference
2. Conditional sequence inferences
alice slipped →
0 1 0 1 she hurt herself, she hurt her knee, ...
1 1 0 1 she fell down, she fell, ...
1 0 1 1 she steadied herself, she caught herself, ...
1 1 1 1 someone caught her, he caught her, ...
0 1 1 1
1 1 1 0
...
k unique codes
1-best per code

Intuition

Learn to generate k different perspectives of the scenario, and condition on these perspectives

COnstrained Decoding with Semantic Sentence Signatures (Cod3s)


(a) alice slipped → she fell down
SBERT
Sentence embedding
Locality-Sensitive Hashing
1 1 0 1
Sentence signature
alice slipped → 1 1 0 1 she fell down
(b) 1. Signature inference 2. Conditional sequence inferences
alice slipped →
k unique codes
0 1 0 1 she hurt herself, she hurt her knee, ...
1 1 0 1 she fell down, she fell, ...
1 0 1 1 she steadied herself, she caught herself, ...
1 1 1 1 someone caught her, he caught her, ...
0 1 1 1
1 1 1 0
...
1-best per code

Intuition

Learn to generate k different perspectives of the scenario, and condition on these perspectives

Advantages

- LSH is a well-studied randomized algorithm; fast implementations are known
- Can fit into training and decoding procedure of Transformers and RNNs

Disadvantages

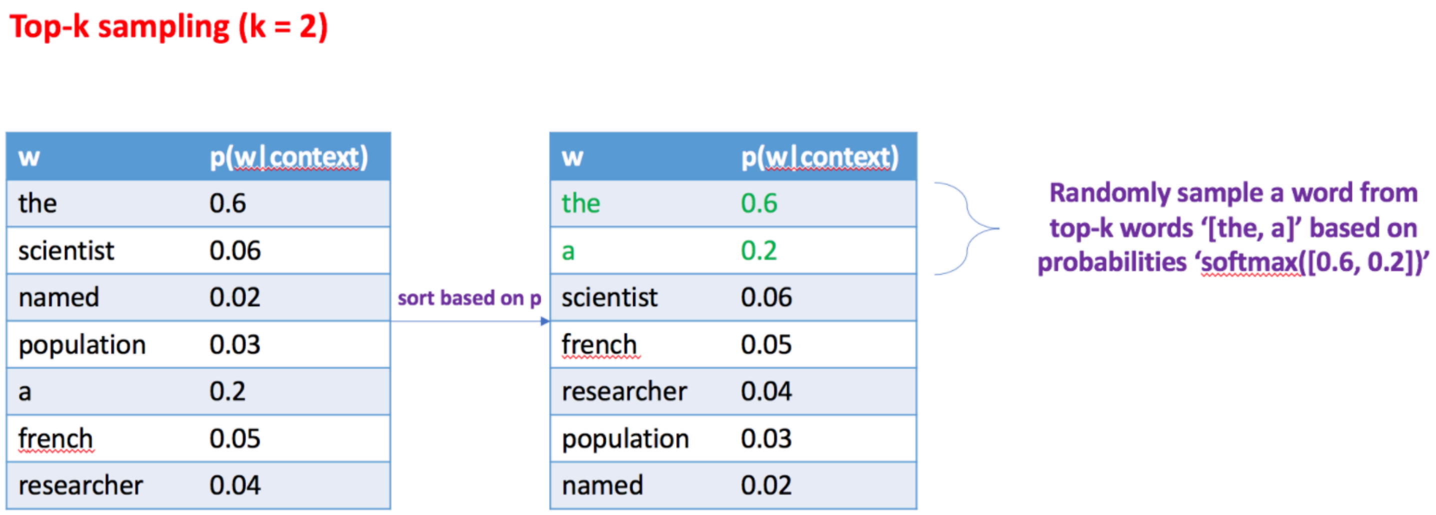
- Two-step inference (signature inference + conditional inference): can't parallelize the steps

Sampling-based diverse generations

1. Top-k sampling
2. Top-p sampling
3. P-exact search (search + sampling)
4. Dynamic beam search (search + sampling)

Top-K Sampling (Fan et al., ACL’18)

Intuition: Suppress unreliable tail by sampling only from top k items



Top-k sampling (k = 2)

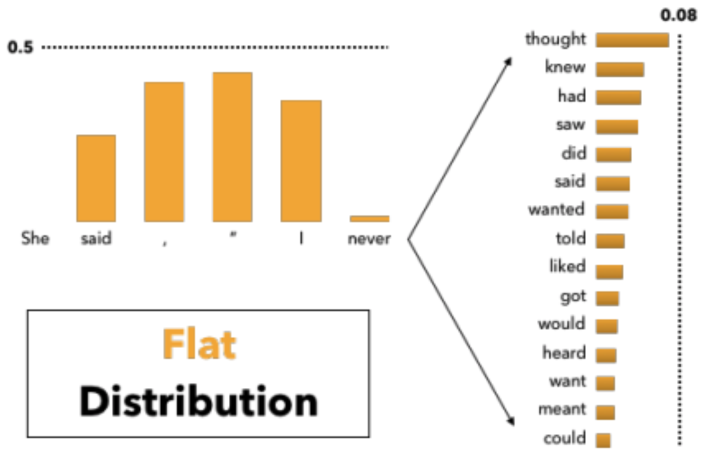
w	p(w|context)
the	0.6
scientist	0.06
named	0.02
population	0.03
a	0.2
french	0.05
researcher	0.04

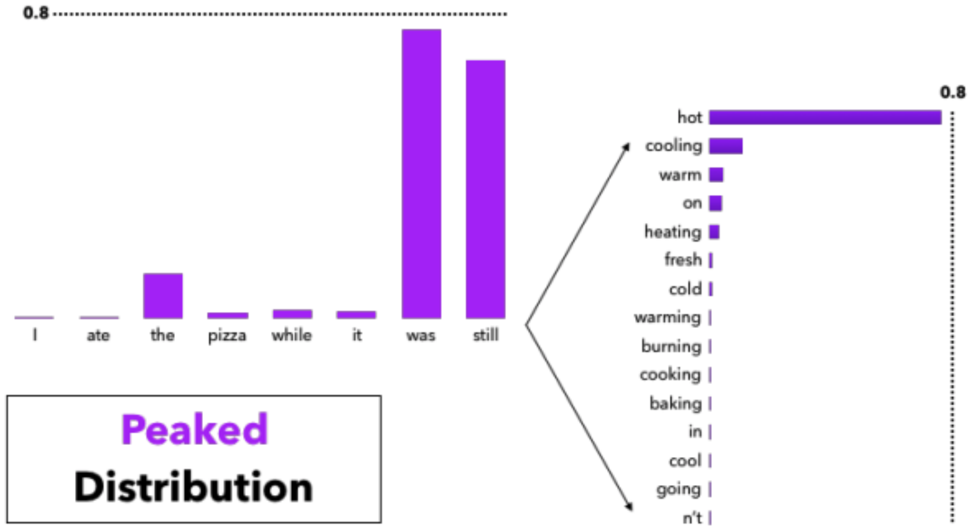
sort based on p

w	p(w|context)
the	0.6
a	0.2
scientist	0.06
french	0.05
researcher	0.04
population	0.03
named	0.02

Randomly sample a word from top-k words ‘[the, a]’ based on probabilities ‘softmax([0.6, 0.2])’

Issue with TOP-K





TOP-P (or NUCLEUS) SAMPLING (Holtzman et al., ICLR'20)

Intuition: Vast majority of probability mass at each time step is concentrated in the nucleus (a small subset of vocabulary)

$$\sum_{x\in V^{(p)}} P(x|x_{1:i-1}) \geq p.$$

$$P'(x|x_{1:i-1}) = \begin{cases} P(x|x_{1:i-1})/p' & \text{if } x \in V^{(p)} \\ 0 & \text{otherwise.} \end{cases}$$

Top-p (nucleus) sampling (p = 0.9)

w	p(w|context)
the	0.6
scientist	0.06
named	0.02
population	0.03
a	0.2
french	0.05
researcher	0.04

sort based on p

w	p(w|context)	cum. p.
the	0.6	0.6
a	0.2	0.8
scientist	0.06	0.86
french	0.05	0.91
researcher	0.04	0.95
population	0.03	0.98
named	0.02	1.0

Randomly sample a word from top-p words '[the, a, scientist, french]' based on probabilities 'softmax([0.6, 0.2, 0.06, 0.05])'

p-EXACT SEARCH (SHAHAM et al., ARXIV'21)

Intuition: Prevent empty sequences and reducing the brevity bias

- Nucleus pruning to round all near-zero probabilities to an absolute zero
- Apply exact search over pruned space, guaranteeing the most probable sequence that contains only top-p tokens at each step
- E.g., P(y_1 = 'George') = 0.567, p = 0.5, renormalize, all probability goes to token 'George'
- E.g., P(y_1 = 'George') = 0.0001, this event is not in the top p, renormalize, prune all sequences beginning with token 'George'

Dynamic Beam Search (SHAHAM et al., ARXIV’21)

Intuition: Increase beam size when entropy is high, prune the number of prefixes when entropy is low

- Generate k_t * IVI candidates
- Normalize probability scores within the candidate set

$$\hat{P}(Y^i) = \frac{P(Y^i)}{\sum_{j=1}^{k_t \cdot |V|} P(Y^j)}$$

- Apply nucleus pruning (p) on the scores
- K_{t+1} = size of nucleus

ADD DIVERSITY TO LOSS

Contrastive SEARCH

Intuition: Underlying reason for model degeneration is the anisotropic distribution of token representations

$$\mathcal{L}_{\mathrm{CL}} = \frac{1}{|\boldsymbol{x}| \times (|\boldsymbol{x}|-1)} \sum_{i=1}^{|\boldsymbol{x}|} \sum_{j=1, j \neq i}^{|\boldsymbol{x}|} \max\{0, \rho - s(h_{x_i}, h_{x_i}) + s(h_{x_i}, h_{x_j})\}$$

$$x_t = \underset{v \in V^{(k)}}{\arg\max} \left\{ (1-\alpha) \times \underbrace{p_\theta(v|\boldsymbol{x}_{<t})}_{\text{model confidence}} - \alpha \times \underbrace{(\max\{s(h_v, h_{x_j}) : 1 \leq j \leq t-1\})}_{\text{degeneration penalty}} \right\}$$

More QUESTIONS?