FALL 2020 CS 395T

TEXAS
The University of Texas at Austin

Neural Decoding

CS 395T: Topics in Natural Language Processing
9/8/2020

Kaj Bostrom and Quang Duong, The University of Texas at Austin

Conditional Text Generation

- Given an input sequence $\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_{i-1}$, generate the next word $\mathbf{w}_i$
- Neural models output a probability distribution over all words **w** in the vocabulary
  - $P(\,\mathbf{w} \mid \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_{i-1}\,)$
- How do we pick $\mathbf{w}_i$?

Option 1: Maximization-Based Decoding

- Find the sequence $\mathbf{w}_i$, $\mathbf{w}_{i+1}$, **…,** $\mathbf{w}_{i+n}$ such that it maximizes the probability over the entire sequence
- Needs to explore every possible sequence
- Large vocabulary sizes make this intractable

Option 2: Greedy Decoding

- Choose the word $\hat{\mathbf{w}}$ with the highest probability given the current sequence to be the next word $\mathbf{w}_i$
- Unsurprisingly, produces low quality text generations

Option 3: Beam Search

- Parameterized by beam width **B**
- Keep **B** sequences of length **i - 1**
- Generate token probabilities for the next word $\mathbf{w}_i$ conditioned on each of these sequences
- Of these, keep the **B** sequences of length **i** with the highest probability
- Degrades to greedy search when **B** = 1

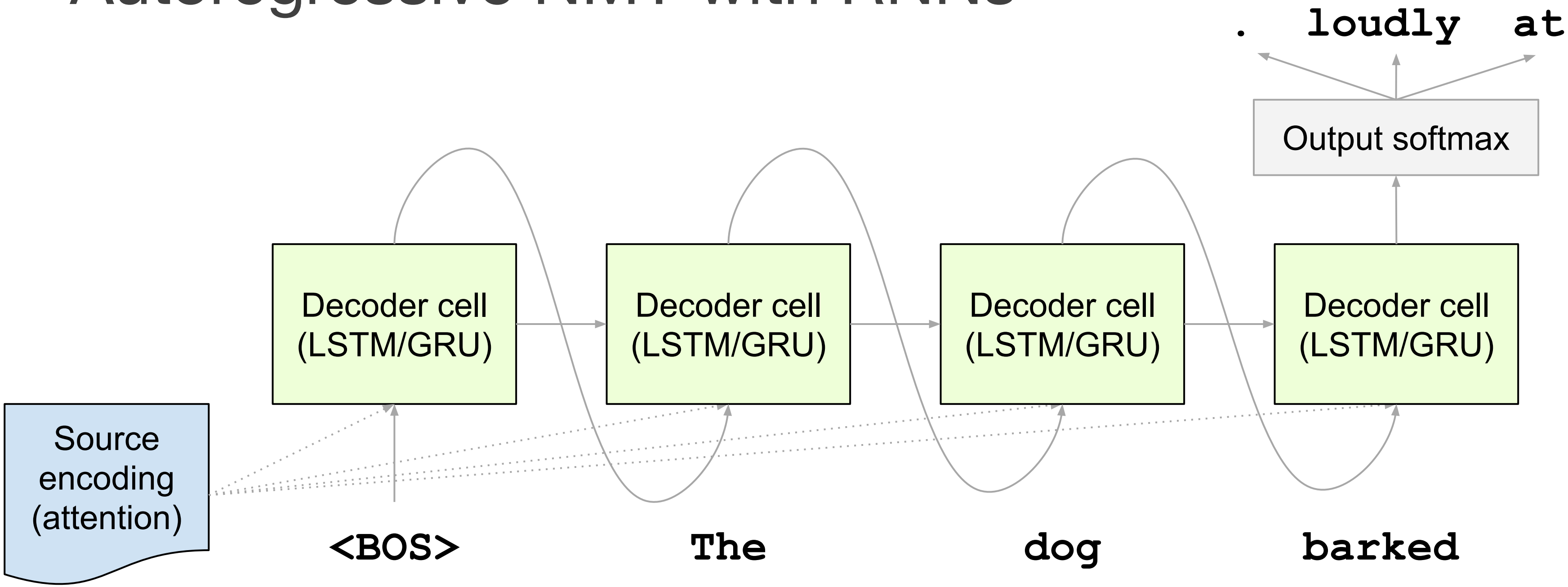
Some Observations

- Classic decoding strategies involve dependencies on words chosen on prior time steps and require sequential inference, which can’t be parallelized on GPUs
- All of them aim to find the highest probability sequence to ensure coherent text is generated
  - How can we produce novel sequences while ensuring the resultant text is still coherent?

Non-Autoregressive Neural Machine Translation [ICLR 2018]

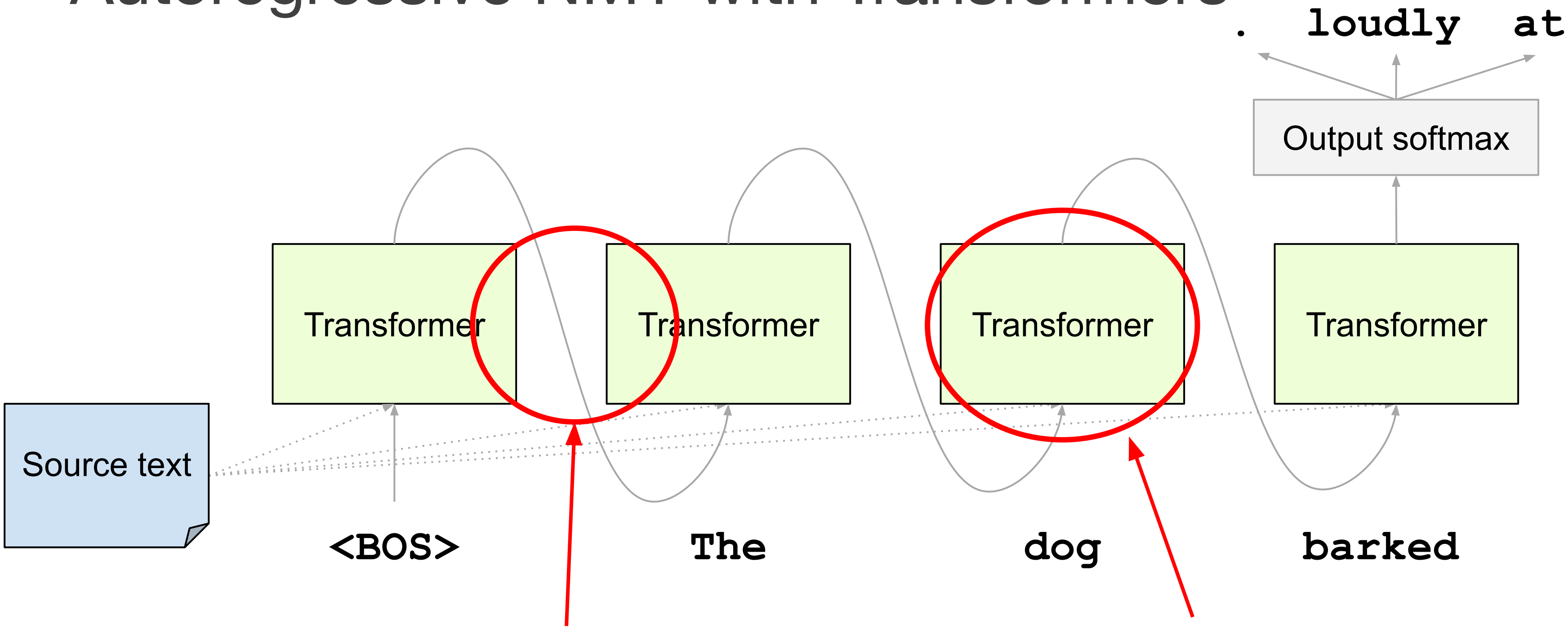
By Jiatao Gu, James Bradbury, Caiming Xiong, Viktor O.K. Li, and Richard Socher

Autoregressive NMT with RNNs



Time complexity linear in sequence length

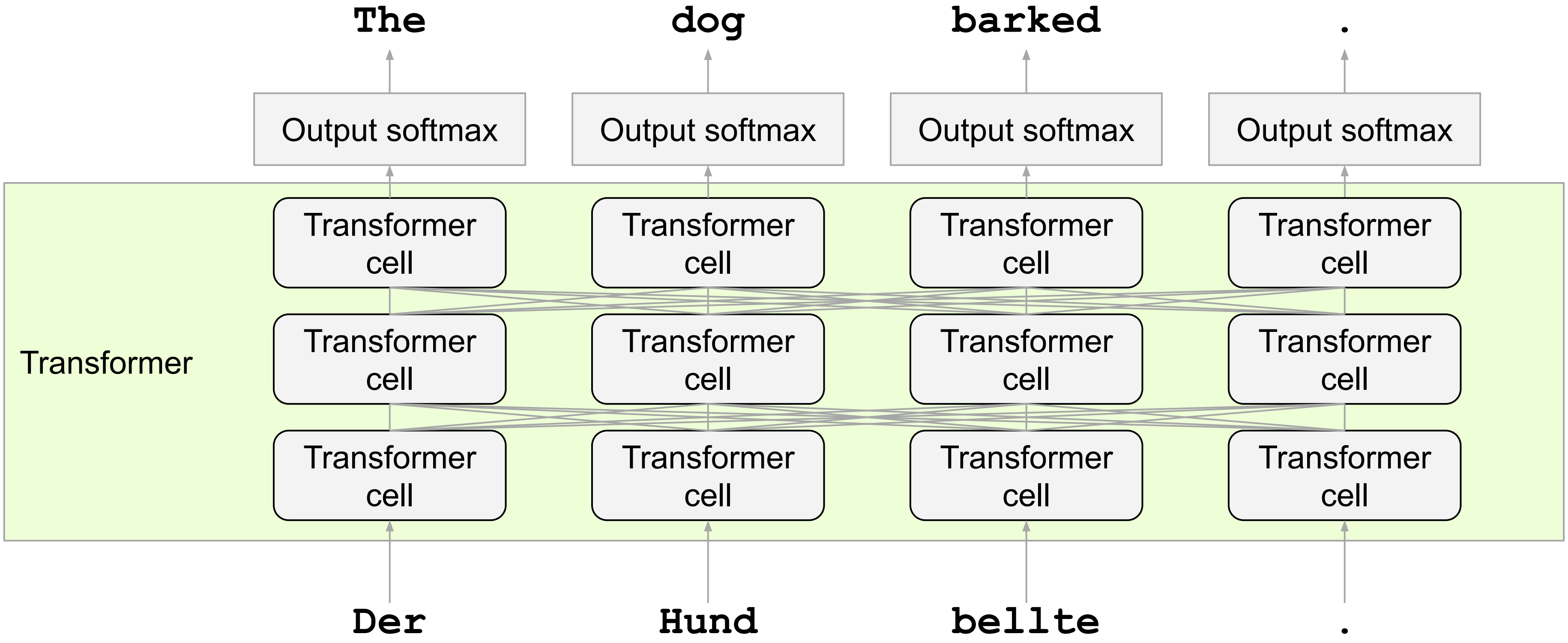
Autoregressive NMT with Transformers



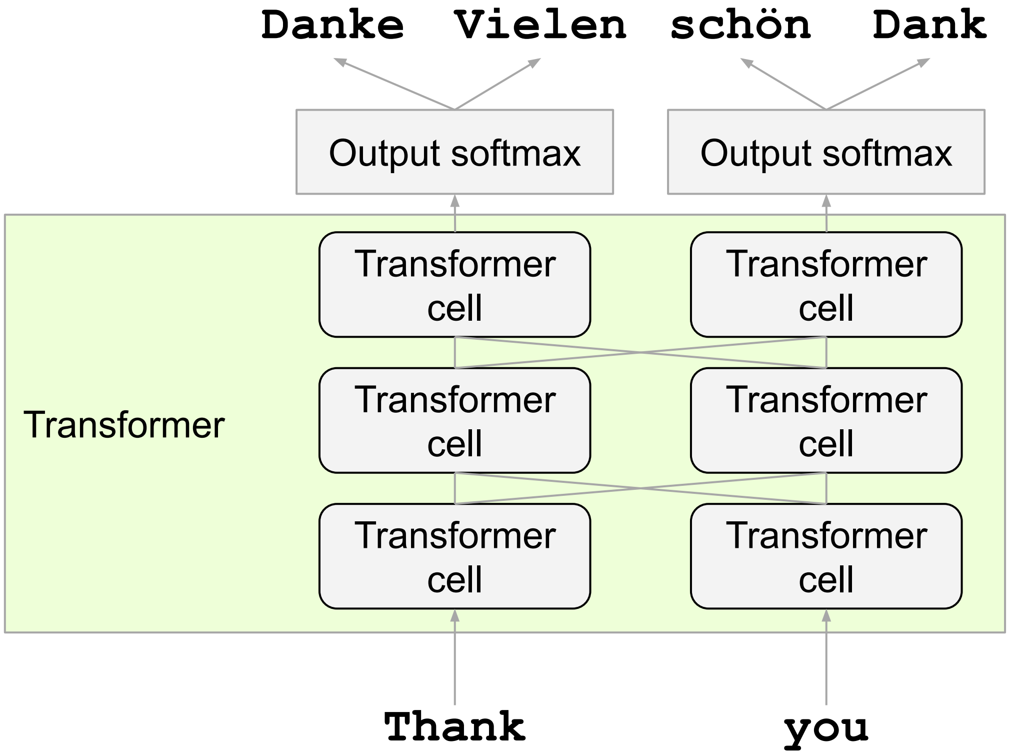
No hidden state dependency between steps, but the next token is still conditioned on previous generation outputs

Applying the entire network at each timestep!

Naïve non-autoregressive (parallel) generation



Problems with the naïve solution



Good:

- “Danke schön”
- “Vielen Dank”

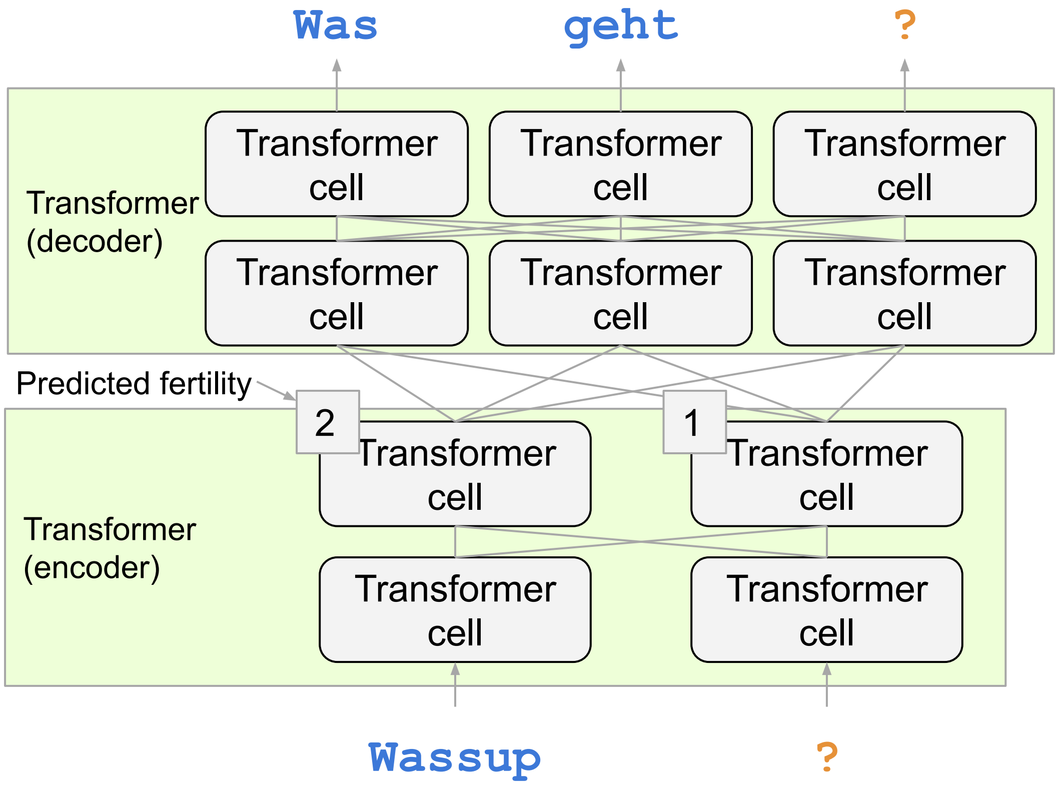
Gibberish:

- “Danke Dank”
- “Vielen schön”

How can we avoid this kind of disagreement without introducing conditioning between generation steps?

Also: how to deal with alignments that aren’t 1 to 1?

Proposed method



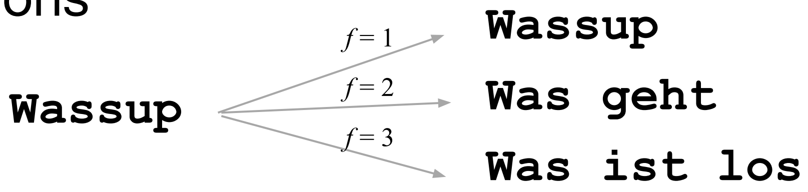
The authors’ proposed solution: Introduce a latent “plan” to hopefully constrain the generation to agree with itself better.

Their proposed latent variable is a sequence of “fertility” values that represent how many target tokens a given source token is aligned with.

The sum of predicted fertility values can also be used to compute the target-side length.

Reasons to use fertility values to guide generation

- Different fertility values can be sampled to explore alternate translations



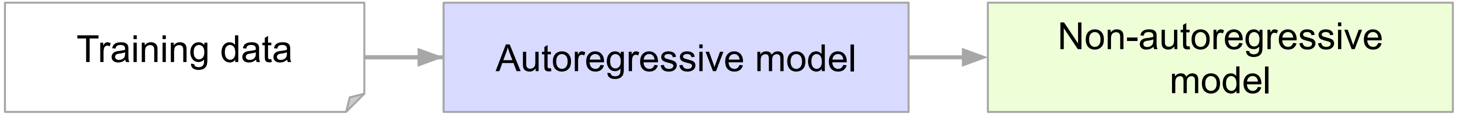
- Fertility values can be supervised at train time using a separate alignment model
- Fertility values are easy to predict and provide a simple, principled way to determine the target sequence length
- Fertility values avoid *entirely* specifying the source → target alignment, which the authors say would place too much modeling burden on the encoder

Problems with fertility values as “generation plans”

Fertility values don’t eliminate word choice ambiguity in all cases:

“Vielen Dank” and “Danke schön” both correspond to the source “Thank you” with fertility values [1, 1]

To get around this, the authors make a compromise:



The autoregressive model’s output is much less diverse than the original training set, so this reduces ambiguity (helping the NAT model) at the cost of data quality.

To recover some of the lost quality, Noisy Parallel Decoding can be applied at the cost of some speed (sampling + reranking using the autoregressive model).

Training with fertility values

True log likelihood

Marginal over fertilities

Prob. of fertilities f given inputs x and params θ

Prob. of target sequence given fertilities f, inputs x and params θ

$$\mathcal{L}_{\text{ML}} = \log p_{\mathcal{NA}}(Y|X;\theta) = \log \sum_{f_{1:T'}\in\mathcal{F}} p_F(f_{1:T'}|x_{1:T'};\theta)\cdot p(y_{1:T}|x_{1:T'}, f_{1:T'};\theta)$$

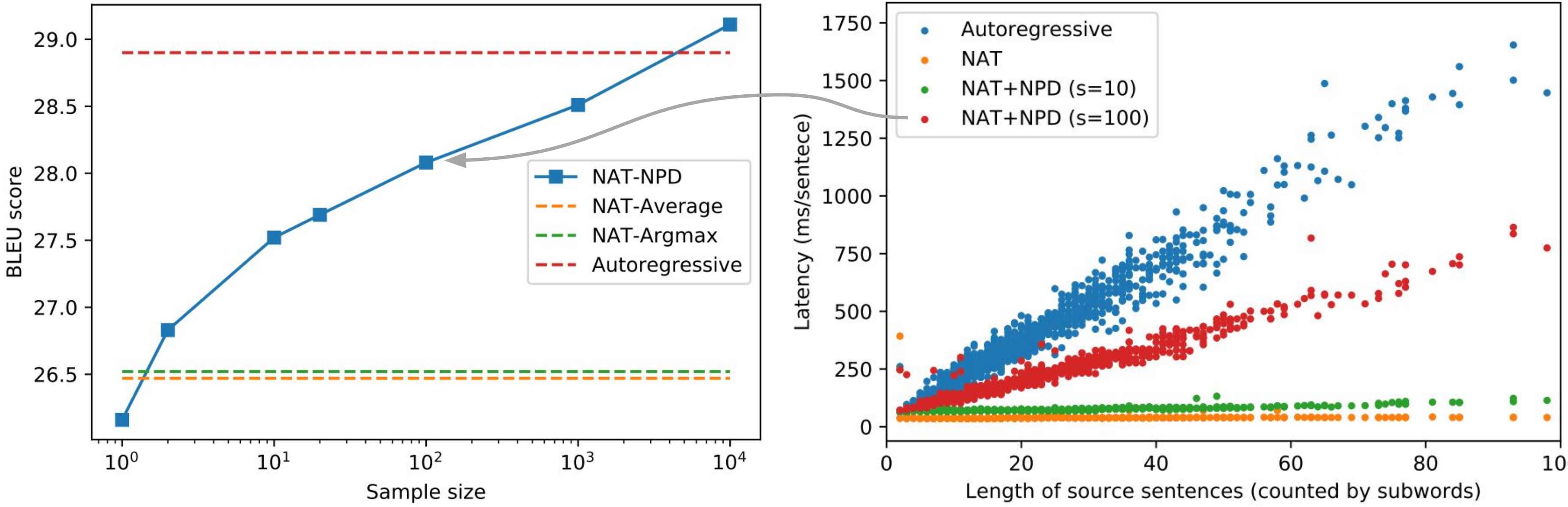
Variational lower bound

$$\geq \mathop{\mathbb{E}}_{f_{1:T'}\sim q}\left(\underbrace{\sum_{t=1}^{T}\log p(y_t|x_1\{f_1\},..,x_{T'}\{f_{T'}\};\theta)}_{\text{Translation Loss}} + \underbrace{\sum_{t'=1}^{T'}\log p_F(f_{t'}|x_{1:T'};\theta)}_{\text{Fertility Loss}}\right) + \mathcal{H}(q)$$

Since we can’t really marginalize over all fertility sequences, we minimize this variational lower bound on the log-likelihood instead.

We minimize the expected log-likelihood over fertility sequences provided by a separate **proposal distribution** q. In practice, this proposal distribution is actually a single fertility sequence computed using either an alignment model or the attention weights of the autoregressive teacher model.

The performance tradeoff summarized



Quality (higher BLEU is better)

Speed (lower latency is better)

Both measured on ISWLT ‘16 dev set (En→De)

(Figures from Gu et al., pages 8 & 12)

Subsequent work on non-autoregressive generation

Latent variable methods:

Ma et al., 2019: FlowSeq: Non-Autoregressive Conditional Sequence Generation with Generative Flow

Flow-based variational model, uses a series of invertible transformations to produce a complex distribution over a continuous latent variable from a simple distribution (a Gaussian)

Shu et al., 2019: Latent-Variable Non-Autoregressive Neural Machine Translation with Deterministic Inference Using a Delta Posterior

Iteratively refine a continuous latent variable (via EM) to maximize a lower bound on the translation log-likelihood at inference time

Subsequent work on non-autoregressive generation

Iterative refinement methods:

Lee et al., 2018: Iterative denoising

Repeatedly apply a model trained to reconstruct corrupted text

Ghazvininejad et al., 2019: Mask-Predict

Mask out & re-predict the least confident output tokens

Stern et al., 2019 (Insertion Transformer), Chan et al., 2019 (KERMIT), and Shen et al., 2020 (Blank LM)

Predict a partial sequence, then fill the remaining gaps in parallel

Mansimov et al., 2020: Gibbs sampling for generation from undirected models

Choose which tokens to predict/re-predict using conditional entropy/confidence; anneal number of predicted tokens per step from L → 1

Subsequent work on non-autoregressive generation

Other methods:

Deng et al., 2020: Cascaded Text Generation with Markov Transformers

Decode a transformer with a Markov context restriction in parallel by iteratively estimating max marginal likelihood + filtering out N-grams at each position (for N=1, then 2, etc…)

Zhou et al., 2020: Understanding Knowledge Distillation in Non-autoregressive Machine Translation

Authors examine why training on AT output is helpful, discovering that repeatedly re-training the AT model on its own output helps even more. They use this to match En→De AT performance with a NAT model.

The Curious Case of Neural Text *De*Generation [ICLR 2020]

By Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi

Directed Text Generation

- Learn the transformation between a given input / output pair of text
- Includes
  - Machine translation
  - Data-to-text generation
  - Summarization

Open-Ended Text Generation

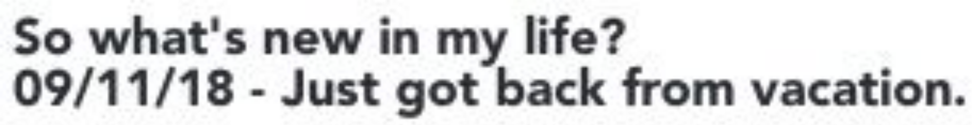
- Learn how to generate text, conditioned on a piece of contextual input
- Includes
  - Conditional Story Generation
  - Contextual Text Continuation

Motivation

- Maximization-based decoding strategies lead to degeneration
  - Incoherent, low-quality text
  - Text loops endlessly
- Especially the case with beam search

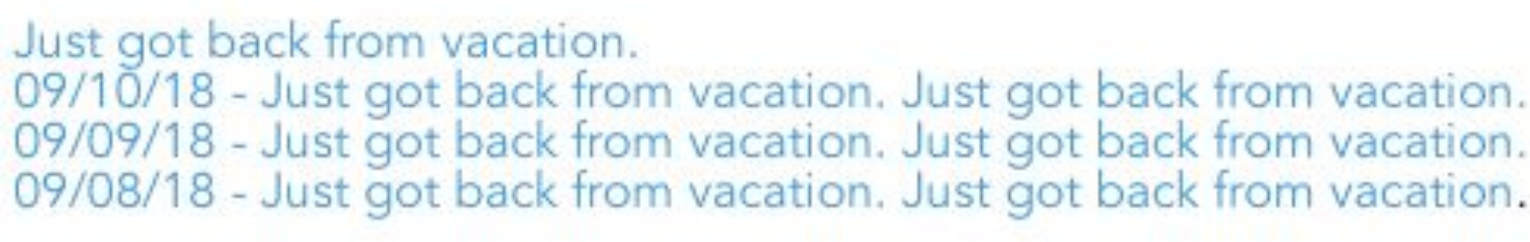
Endless Looping

Context:



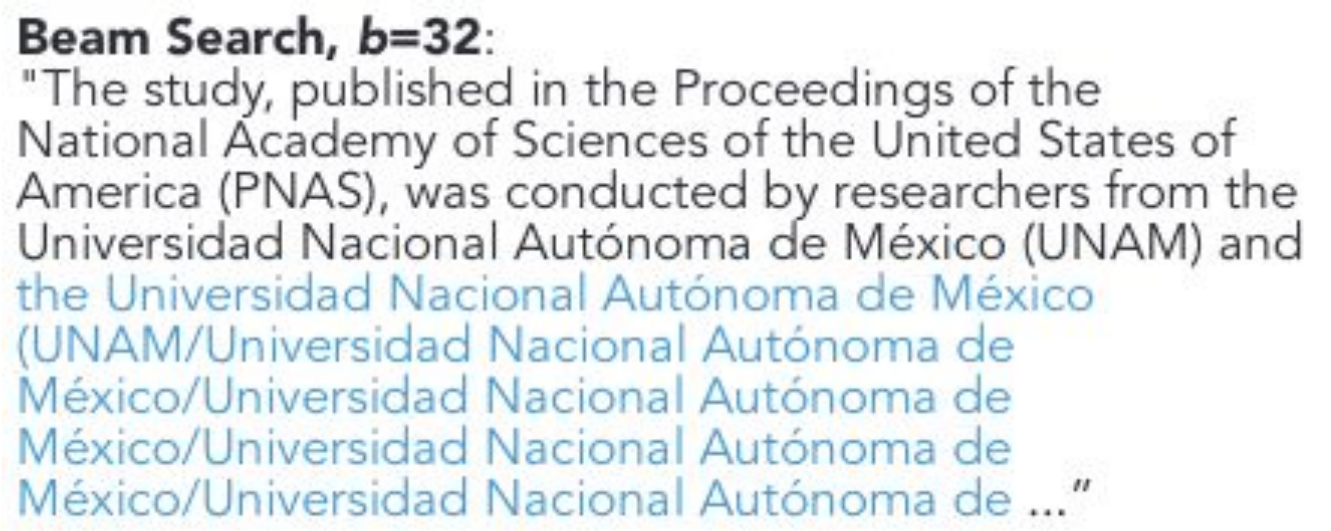
So what's new in my life?
09/11/18 - Just got back from vacation.

Beam Search Produces:



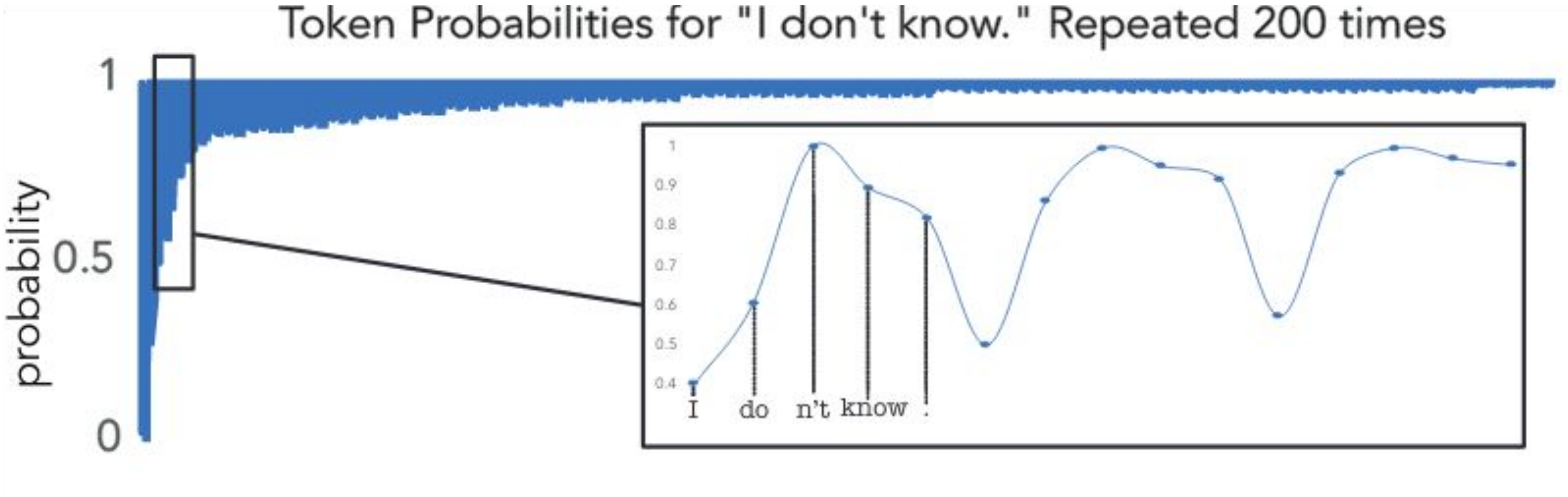
Just got back from vacation.
09/10/18 - Just got back from vacation. Just got back from vacation.
09/09/18 - Just got back from vacation. Just got back from vacation.
09/08/18 - Just got back from vacation. Just got back from vacation.

Endless Looping



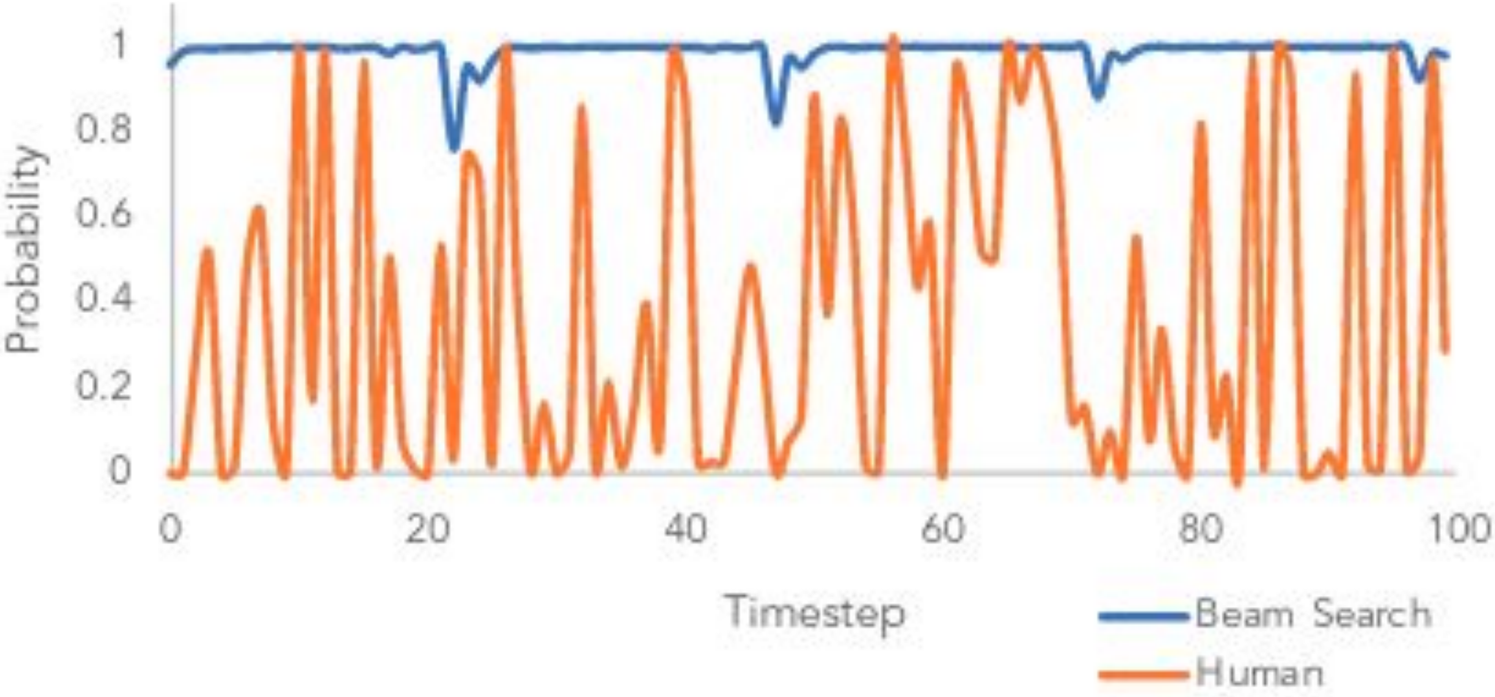
Beam Search, *b*=32:
"The study, published in the Proceedings of the National Academy of Sciences of the United States of America (PNAS), was conducted by researchers from the Universidad Nacional Autónoma de México (UNAM) and the Universidad Nacional Autónoma de México (UNAM/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de México/Universidad Nacional Autónoma de ..."

Positive Feedback Loop

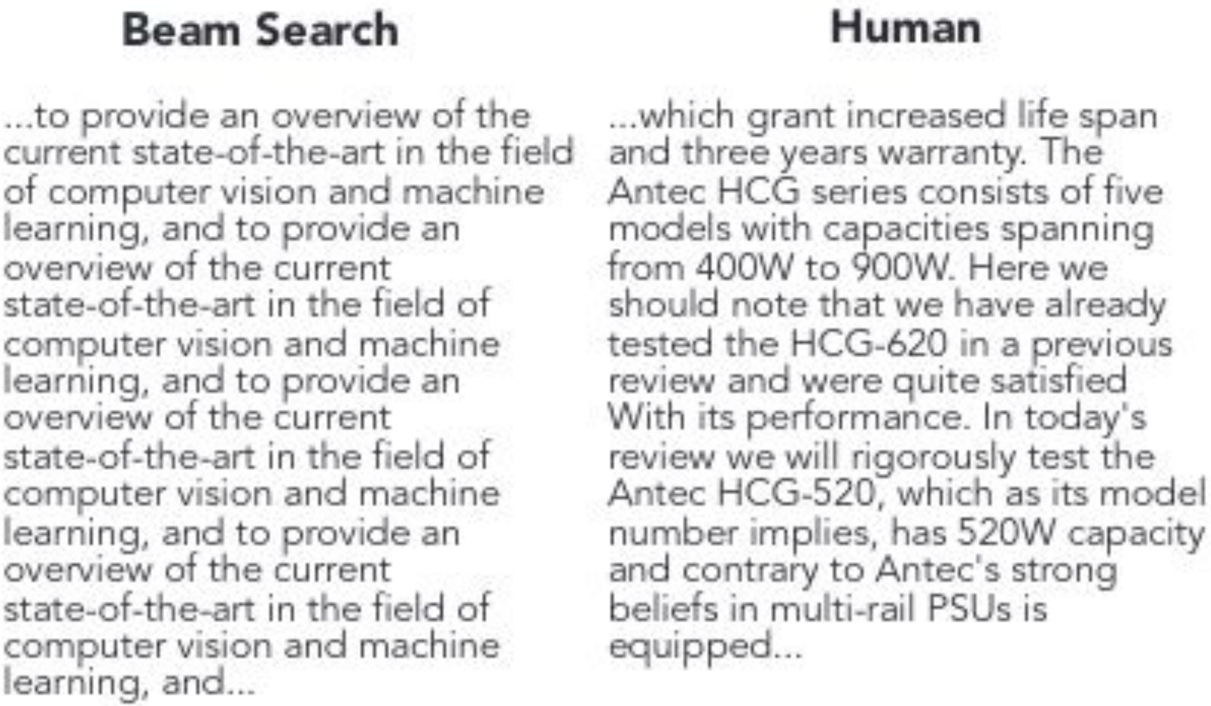


Token Probabilities for "I don't know." Repeated 200 times

Beam Search Text is Less Surprising



“Beam Search Text is Less Surprising”



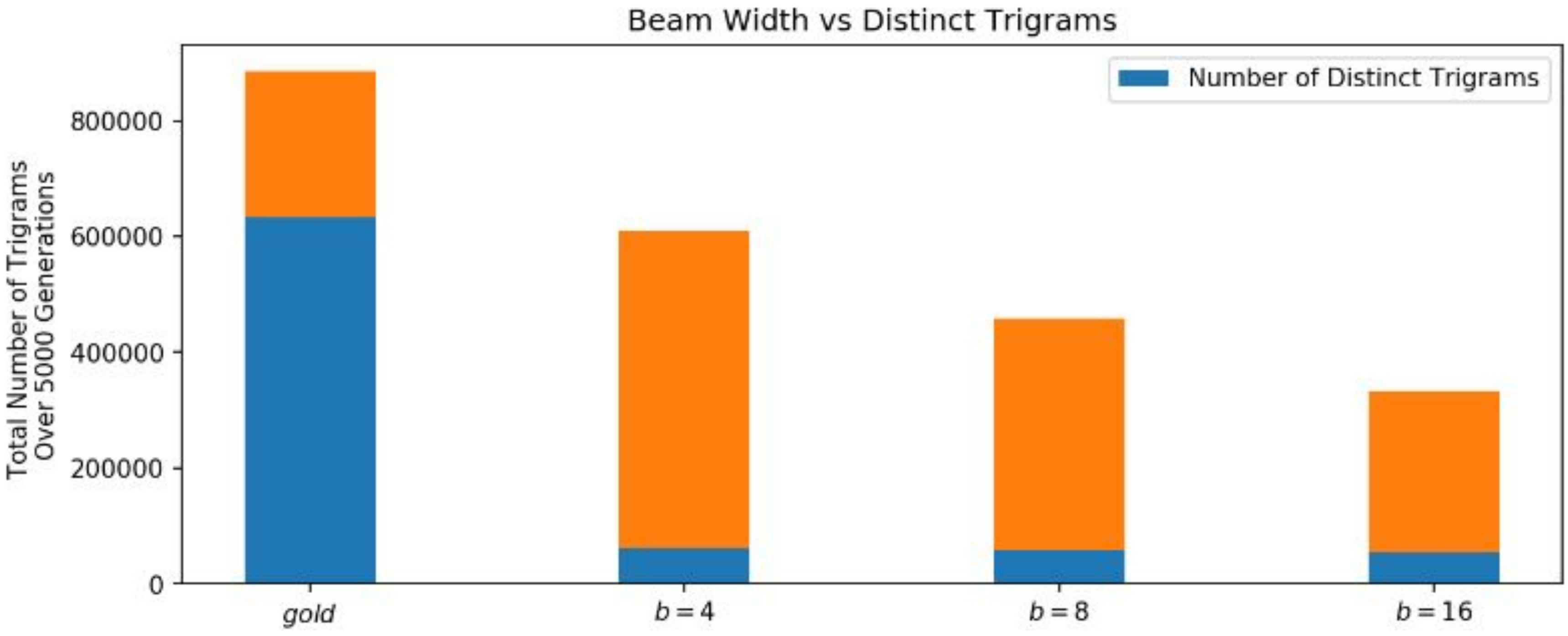
Beam Search

...to provide an overview of the current state-of-the-art in the field of computer vision and machine learning, and to provide an overview of the current state-of-the-art in the field of computer vision and machine learning, and to provide an overview of the current state-of-the-art in the field of computer vision and machine learning, and to provide an overview of the current state-of-the-art in the field of computer vision and machine learning, and...

Human

...which grant increased life span and three years warranty. The Antec HCG series consists of five models with capacities spanning from 400W to 900W. Here we should note that we have already tested the HCG-620 in a previous review and were quite satisfied With its performance. In today's review we will rigorously test the Antec HCG-520, which as its model number implies, has 520W capacity and contrary to Antec's strong beliefs in multi-rail PSUs is equipped...

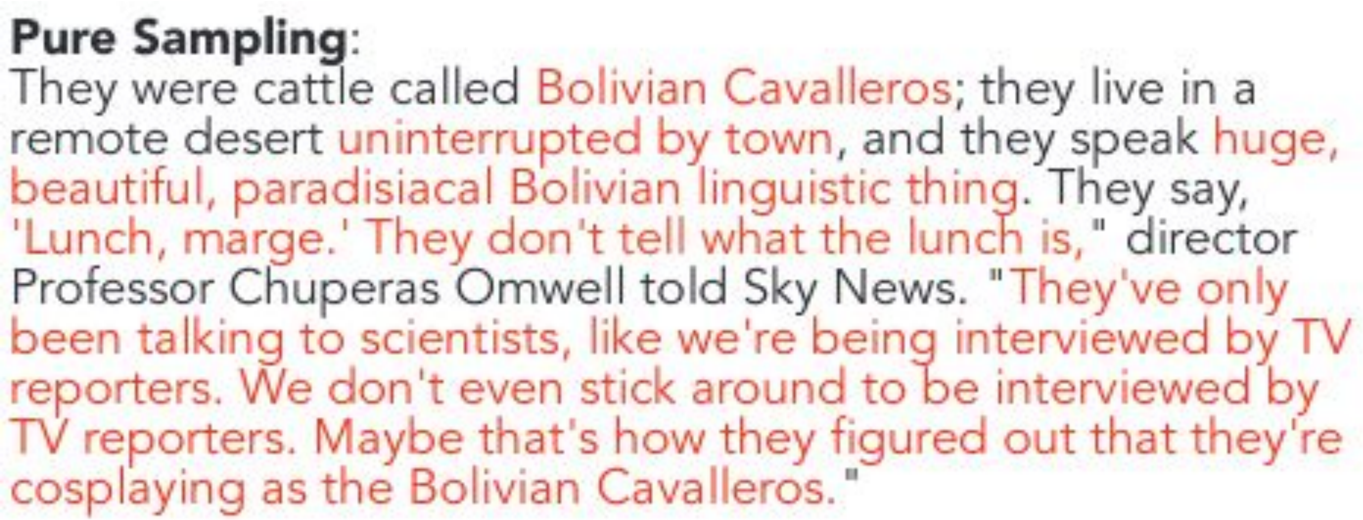
Beam Search


Beam Width vs Distinct Trigrams
Number of Distinct Trigrams
Total Number of Trigrams
Over 5000 Generations
0
200000
400000
600000
800000
gold
$b = 4$
$b = 8$
$b = 16$

Pure Sampling

- Sample directly from the model’s outputted probabilities
- Produces low-quality, incoherent text due to “unreliable tail” of distribution

Pure Sampling



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."

Sampling with Temperature

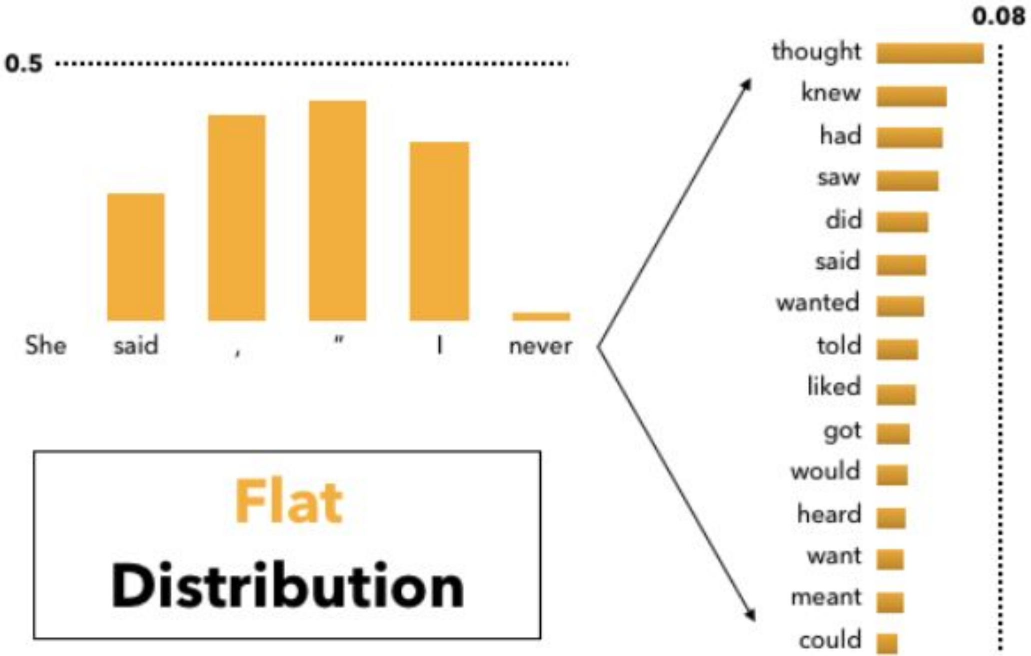
- Temperatures between 0 and 1 skew the probability distribution towards higher probability words
- Lower temperatures reduce diversity

$$p(x = V_l | x_{1:i-1}) = \frac{\exp(u_l/t)}{\sum_{l'} \exp(u'_l/t)}$$

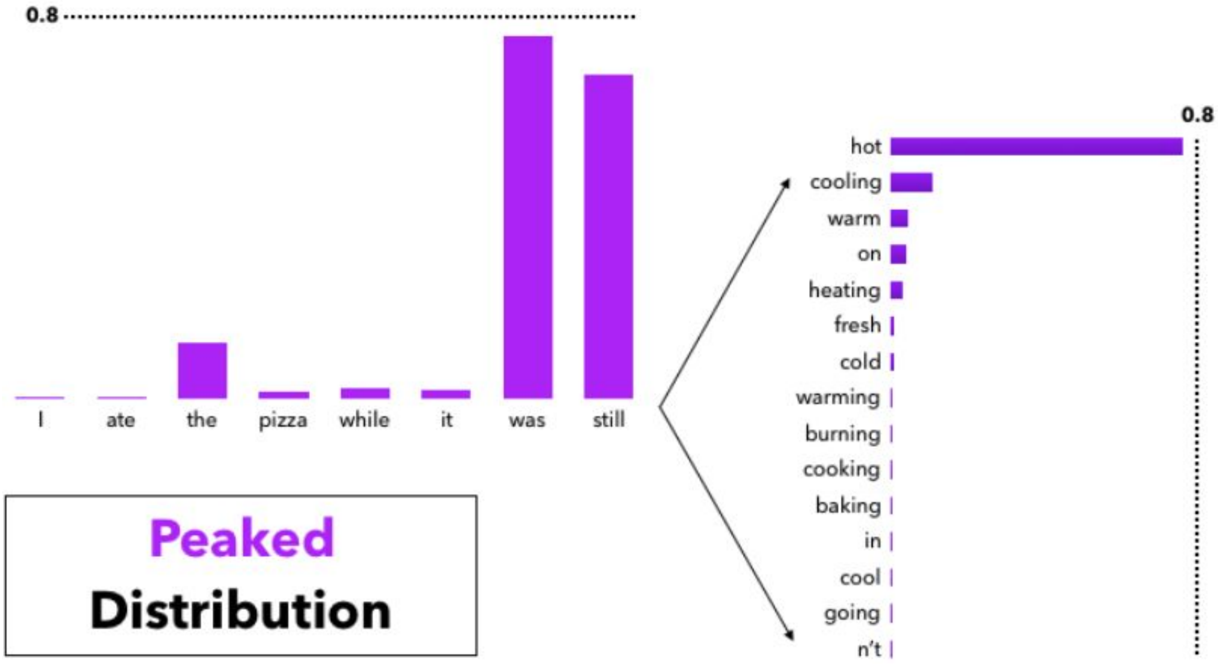
Top-K Sampling

- Sample from the K highest probability words at each time step
- Difficult to pick a good K because of different probability distribution shapes
- [Fan 2018]

Top-K Sampling (Larger K Better)



Top-K Sampling (Smaller K Better)



Nucleus Sampling

- For a given probability p, the top-p vocabulary is the smallest set such that

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

- Size of vocabulary adjusts with shape of the language model’s probability distribution

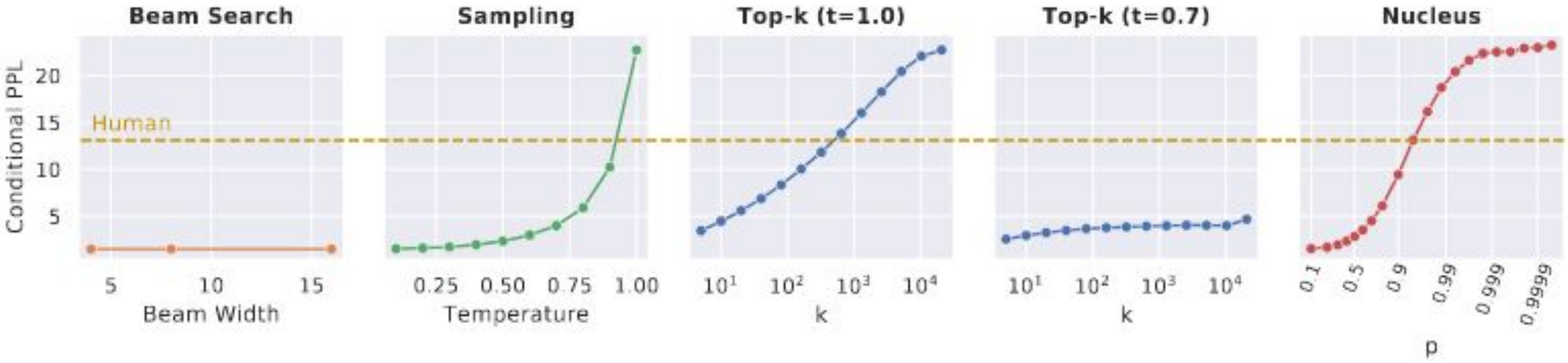
Experimental Setup

- GPT2 Large (762M Params) [Radford 2019]
- Trained on WebText (40GB of web scraped text)
- Generate 5000 text passages until end-of-document token generated or 200 tokens
- Conditioned on 1-40 token long paragraph held-out of WebText

Perplexity

- Measures how surprising or unexpected a piece of text is relative to a language model
- Lower perplexity has generally been seen as a good thing, but maximization-based decoding leads to lower perplexities than seen in human text

Perplexity

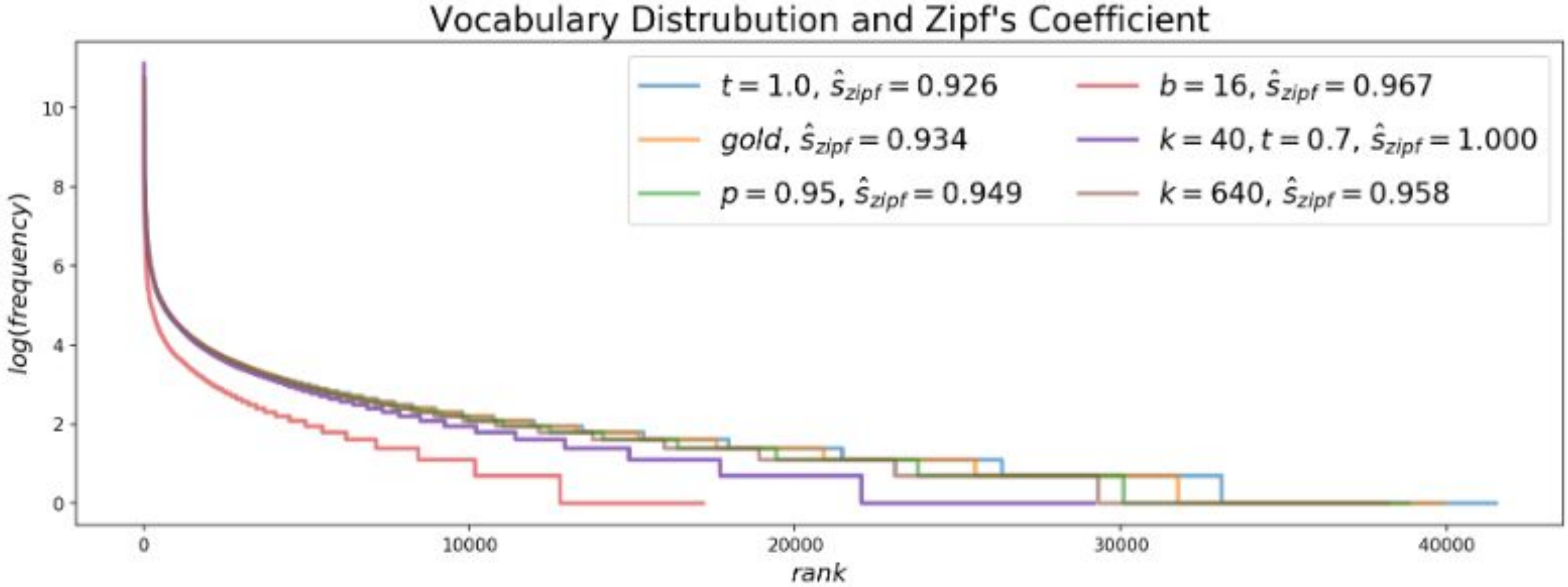

Beam Search
Sampling
Top-k (t=1.0)
Top-k (t=0.7)
Nucleus
Conditional PPL
20
15
10
5
Human
5
10
15
Beam Width
0.25
0.50
0.75
1.00
Temperature
10^1
10^2
10^3
10^4
k
10^1
10^2
10^3
10^4
k
0.1
0.5
0.9
0.99
0.999
0.9999
p

Perplexity

- Beam search has perplexities far below human text
- Top-K, temperature, and nucleus sampling can reach human levels
    - Frequently used parameter settings for Top-K and temperature sampling would place it below human perplexity

Zipf’s Law and Zipfian Distribution

- Exponential relationship between rank of a word and frequency in text
- $\hat{s}_{zipf}$ is the Zipfian coefficient and can be compared to a perfect exponential relationship where $\hat{s}_{zipf} = 1$

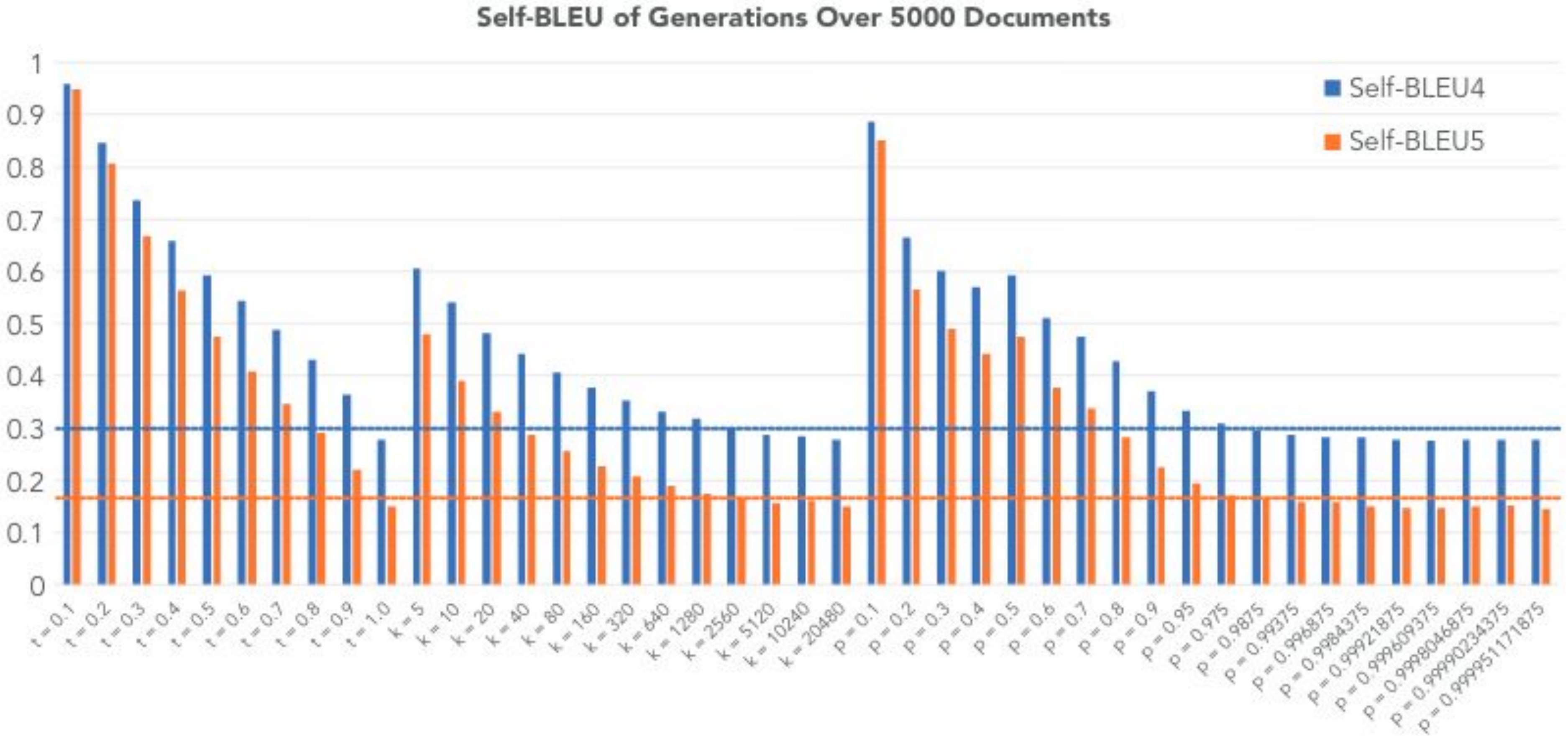

Vocabulary Distrubution and Zipf's Coefficient
$t = 1.0$, $\hat{s}_{zipf} = 0.926$
$gold$, $\hat{s}_{zipf} = 0.934$
$p = 0.95$, $\hat{s}_{zipf} = 0.949$
$b = 16$, $\hat{s}_{zipf} = 0.967$
$k = 40, t = 0.7$, $\hat{s}_{zipf} = 1.000$
$k = 640$, $\hat{s}_{zipf} = 0.958$
log(frequency)
10
8
6
4
2
0
0
10000
20000
30000
40000
rank

Zipf’s Law and Zipfian Distribution

- Pure sampling and nucleus sampling are closest to the target human distribution
- Pure sampling overestimates the prevalence of rarer (larger rank) words

Self-BLEU

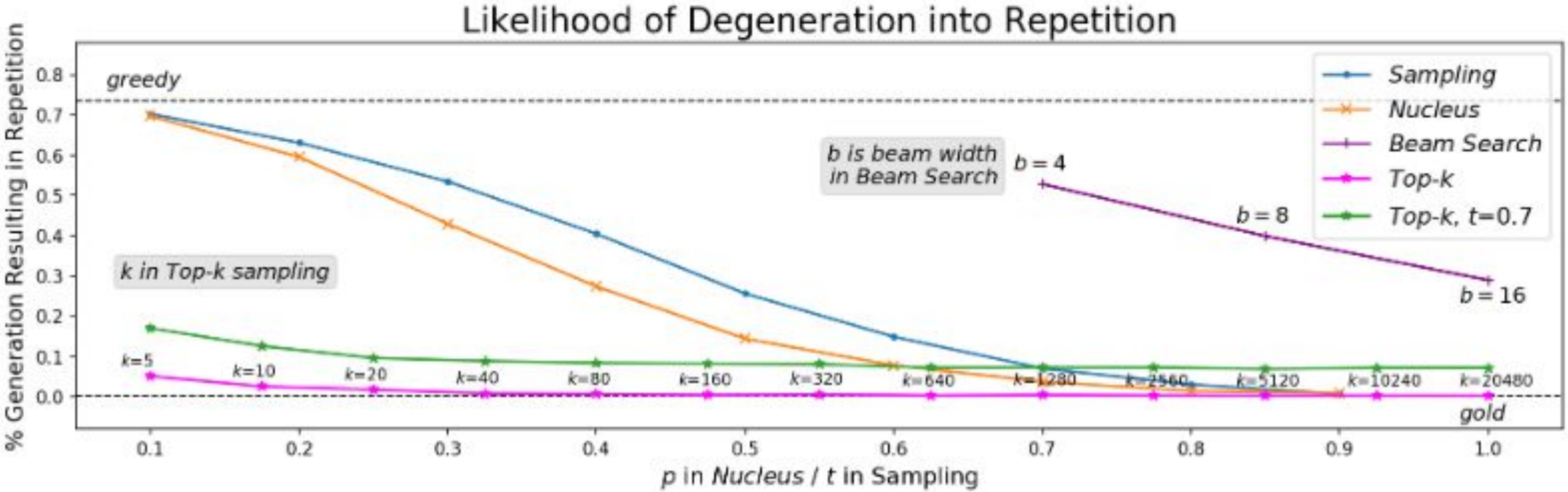
- Compute BLEU score with other generations from the same decoding strategy as references
- Lower score indicates higher diversity as they have fewer n-grams in common
- Sampled 1000 from the 5000 total generations and compared against the remaining 4999
- [Zhu et al. 2018]



Self-BLEU of Generations Over 5000 Documents

Self-BLEU

- Commonly used values of temperature and k have higher Self-BLEU scores, and therefore lower diversity
- High values of temperature and k have higher diversity, but perplexity soars unnaturally high
- Reasonable values of p for nucleus sampling are near human levels of Self-BLEU scores


Likelihood of Degeneration into Repetition
% Generation Resulting in Repetition
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
greedy
gold
Sampling
Nucleus
Beam Search
Top-k
Top-k, t=0.7
b is beam width in Beam Search
b = 4
b = 8
b = 16
k in Top-k sampling
k=5
k=10
k=20
k=40
k=80
k=160
k=320
k=640
k=1280
k=2560
k=5120
k=10240
k=20480
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1.0
p in Nucleus / t in Sampling

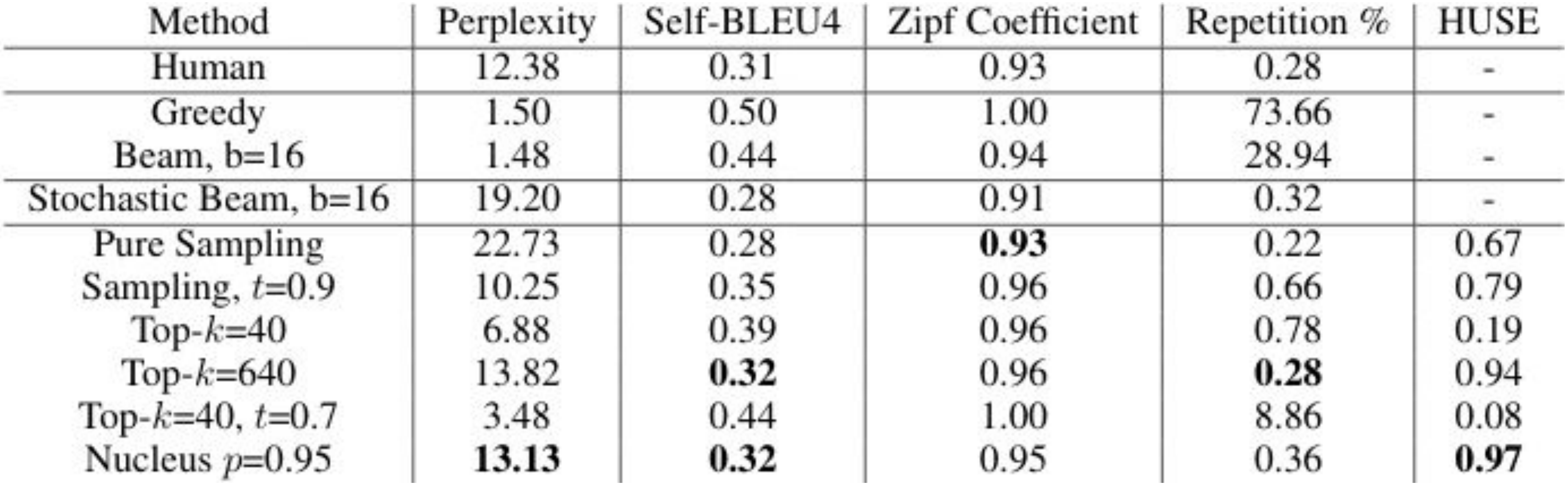
Human Unified with Statistical Evaluation

- HUSE scores combine human judgement of quality with statistical evaluation of diversity
- 200 generations for each of the decoding technique (20 annotations each from 20 different annotators)
- Truncated probability distributions received a 0, so they were interpolated to produce a smoother distribution
- [Hashimoto 2019]

Metrics in Review

- **Perplexity**: Measures how surprising the text is and its general coherence
- **Self-BLEU**: Measures generated diversity
- **Zipf Coefficient**: Measures closeness to exponential relationship for word rank to frequency
- **Repetition %**: Measures amount of repeated text
- **HUSE**: Measures quality and diversity of text

Results



Method	Perplexity	Self-BLEU4	Zipf Coefficient	Repetition %	HUSE
Human	12.38	0.31	0.93	0.28	-
Greedy	1.50	0.50	1.00	73.66	-
Beam, b=16	1.48	0.44	0.94	28.94	-
Stochastic Beam, b=16	19.20	0.28	0.91	0.32	-
Pure Sampling	22.73	0.28	**0.93**	0.22	0.67
Sampling, t=0.9	10.25	0.35	0.96	0.66	0.79
Top-k=40	6.88	0.39	0.96	0.78	0.19
Top-k=640	13.82	**0.32**	0.96	**0.28**	0.94
Top-k=40, t=0.7	3.48	0.44	1.00	8.86	0.08
Nucleus p=0.95	**13.13**	**0.32**	0.95	0.36	**0.97**

Results

- Nucleus Sampling is closest to human generated text for most of the metrics
- Common parameter settings for Beam Search and Top-K / Temperature Sampling results in significant amounts of repetition, low diversity, and lower-than-human perplexity

Related Work

- GAN generated text [Yu 2018; Xu 2019]
    - Generations worse than those from LMs [Caccia 2018; Tevet 2019; Semenuita 2020]
- Constraining beam search via a diversity scoring function or by requiring diversity in beam hypotheses [Li 2016a; Vijayakumar 2018; Kulikov 2019; Pal 2006]
- “Unlikelihood Loss” reduces loss and therefore the gradient signal for repeated tokens [Welleck 2020]

What this paper did well

- Deep dive into where current decoding strategies fail and proposed an intuitive and simple alternative backed by strong empirical evidence
- Both Top-K and Nucleus sampling has been used in recent work, but mostly in language modeling and (conditioned) language generation:
  - CTRL: A Conditional Transformer Model for Controllable Generation [Keskar 2019]
  - Defending Against Neural Fake News [Zellers 2019]
  - Language Models are Unsupervised Multitask Learners [Radford 2019]

Where it could do better

- Proposed technique is effectively a re-imagining of Top-K Sampling which can perform about the same
- Does not compare with other techniques used to combat degeneration like the diverse/constrained beam search and the “unlikelihood loss”
- Both Top-K and Nucleus Sampling criticized in [Welleck 2020] for not resolving the underlying problem of poor quality token-level probabilities

YOU
BELONG
HERE

CNS Diversity and Inclusion • cns.utexas.edu/diversity

Questions?