LLMs for code (4)

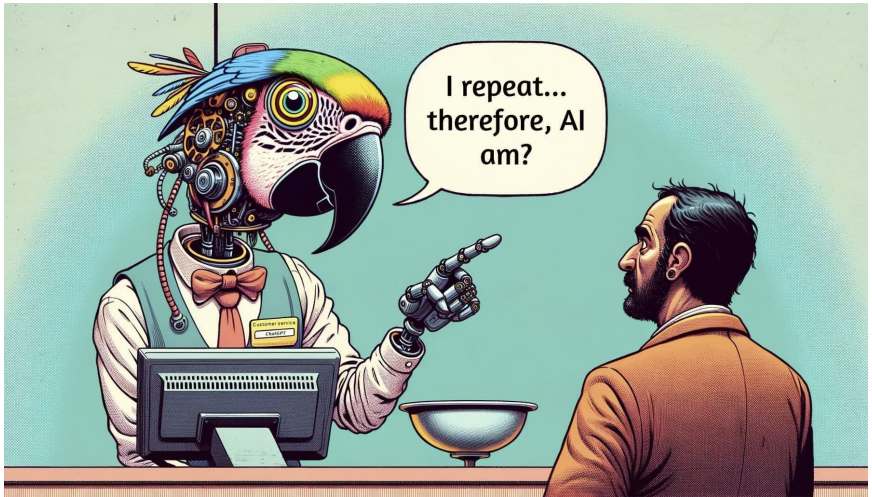
Marc LELARGE
INRIA-ENS
Paris
Co-teacher: Nathanaël FIJALKOW

Token-level generation

Make observed data likely under the model: maximum likelihood

$$\arg\max_{\theta} \frac{1}{|\mathcal{D}|} \sum_{(y_1,\ldots,y_\ell)\in\mathcal{D}} \log p_\theta(y_1,\ldots,y_\ell) = \arg\max_{\theta} \frac{1}{|\mathcal{D}|} \sum_{(y_1,\ldots,y_\ell)\in\mathcal{D}} \sum_{t=1}^{\ell} \log p_\theta(y_t|y_{<t})$$

Now, how to generate tokens with our model?

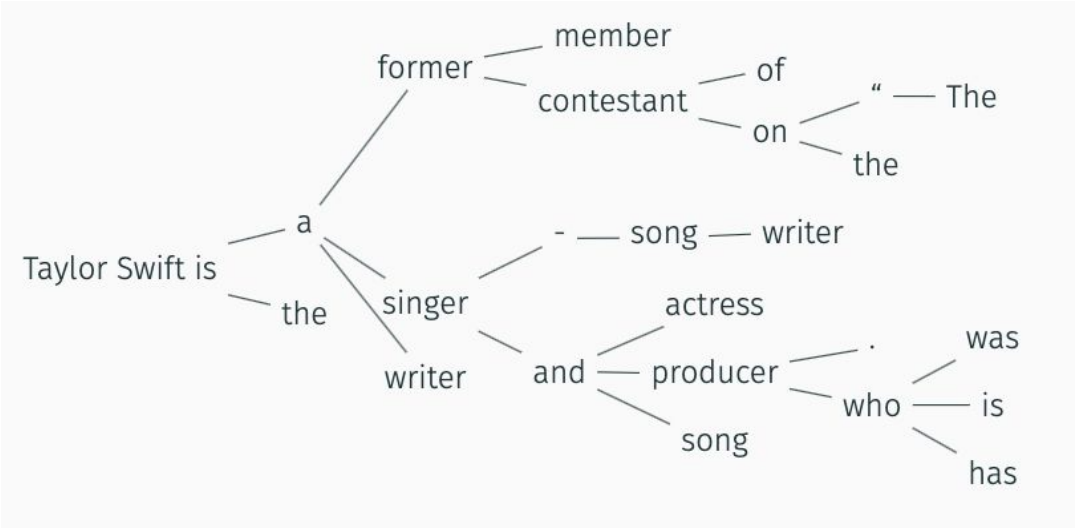


Decoding as optimization

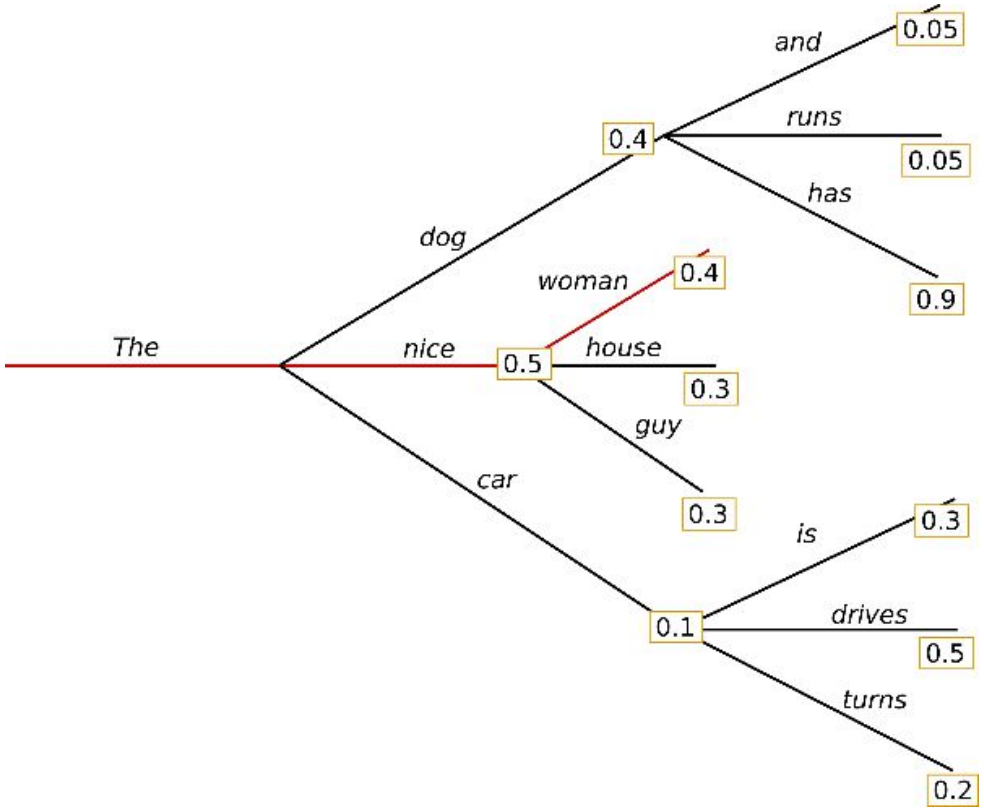
Maximum a posteriori (MAP) decoding: given a prompt x, solve:

$$\arg \max_{(y_1,\ldots,y_\ell)} \log p_\theta(y_1,\ldots,y_\ell|x) = \arg \max_{(y_1,\ldots,y_\ell)} \sum_{t=1}^{\ell} \log p_\theta(y_t|y_{<t},x)$$

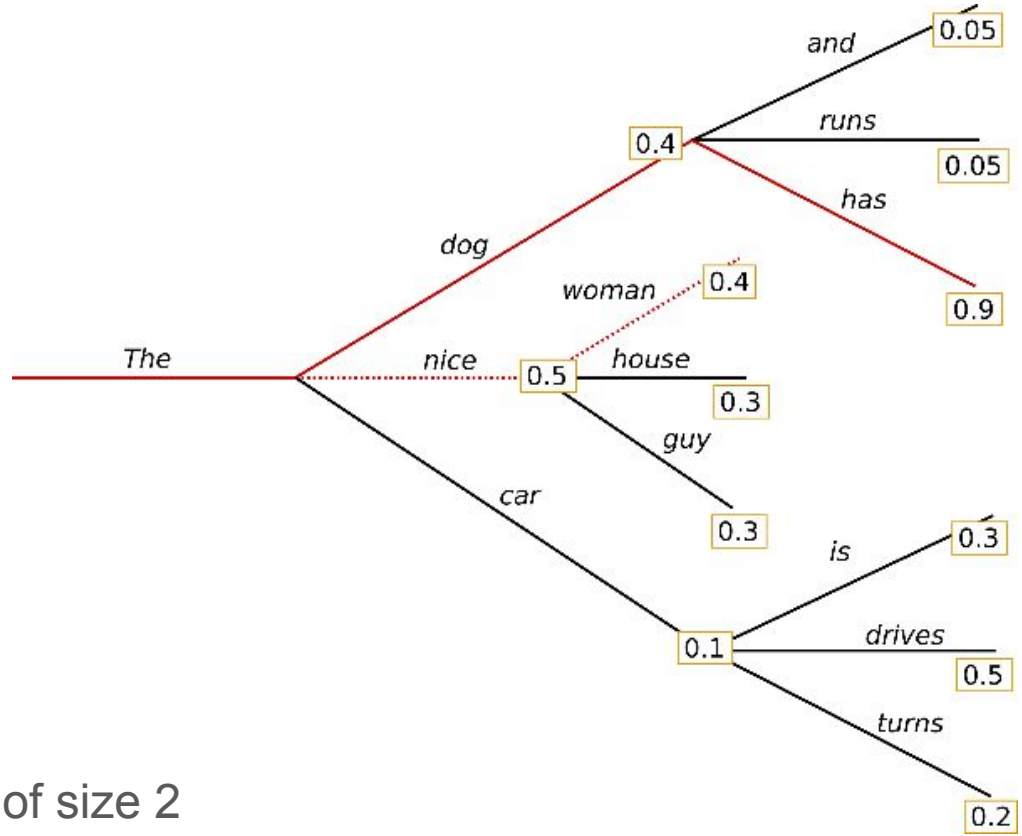
This is a search problem
with a large branching factor
-> too hard to solve exactly…



Greedy decoding



Beam search



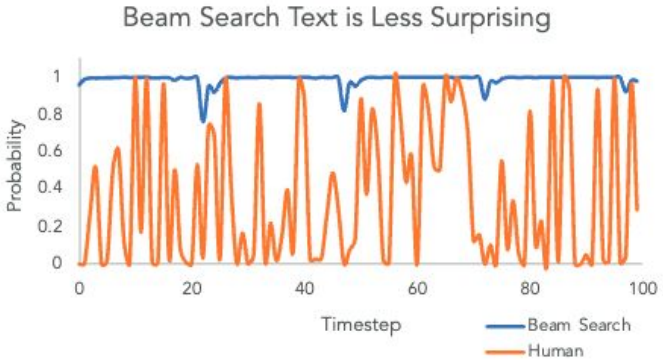
Ex with beam of size 2

Beam search in code

```
def step(
    self, step, lprobs, scores, prev_output_tokens=None, original_batch_idxs=None
):
    """Take a single search step.

    Args:
        step: the current search step, starting at 0
        lprobs: (bsz x input_beam_size x vocab_size)
            the model's log-probabilities over the vocabulary at the current step
        scores: (bsz x input_beam_size x step)
            the historical model scores of each hypothesis up to this point
        prev_output_tokens: (bsz x step)
            the previously generated oputput tokens
        original_batch_idxs: (bsz)
            the tensor with the batch indices, in the range [0, bsz)
            this is useful in case there has been applied a re-ordering
            and we need to know the orignal indices

    Return: A tuple of (scores, indices, beams) where:
        scores: (bsz x output_beam_size)
            the scores of the chosen elements; output_beam_size can be
            larger than input_beam_size, e.g., we may return
            2*input_beam_size to account for EOS
        indices: (bsz x output_beam_size)
            the indices of the chosen elements
        beams: (bsz x output_beam_size)
            the hypothesis ids of the chosen elements, in the range [0, input_beam_size)
    """
    raise NotImplementedError
```

```
class BeamSearch(Search):

    @torch.jit.export
    def step(
        self,
        step: int,
        lprobs,
        scores: Optional[Tensor],
        prev_output_tokens: Optional[Tensor] = None,
        original_batch_idxs: Optional[Tensor] = None,
        candidate_multiple: int = 2,
    ):
        bsz, beam_size, vocab_size = lprobs.size()

        if step == 0:
            # at the first step all hypotheses are equally likely, so use
            # only the first beam
            lprobs = lprobs[:, ::beam_size, :].contiguous()
        else:
            # make probs contain cumulative scores for each hypothesis
            assert scores is not None
            lprobs = lprobs + scores[:, :, step - 1].unsqueeze(-1)

        top_prediction = torch.topk(
            lprobs.view(bsz, -1),
            k=min(
                # Take the best `candidate_muliple`(default 2) x beam_size predictions. We'll choose the first
                # beam_size of these which don't predict eos to continue with.
                candidate_multiple * beam_size,
                lprobs.view(bsz, -1).size(1) - 1,  # -1 so we never select pad
            ),
        )
        scores_buf = top_prediction[0]
        indices_buf = top_prediction[1]
        # Project back into relative indices and beams
        beams_buf = torch.div(indices_buf, vocab_size, rounding_mode="trunc")
        indices_buf = indices_buf.fmod(vocab_size)

        # At this point, beams_buf and indices_buf are single-dim and contain relative indices
        return scores_buf, indices_buf, beams_buf
```

Problems with beam search



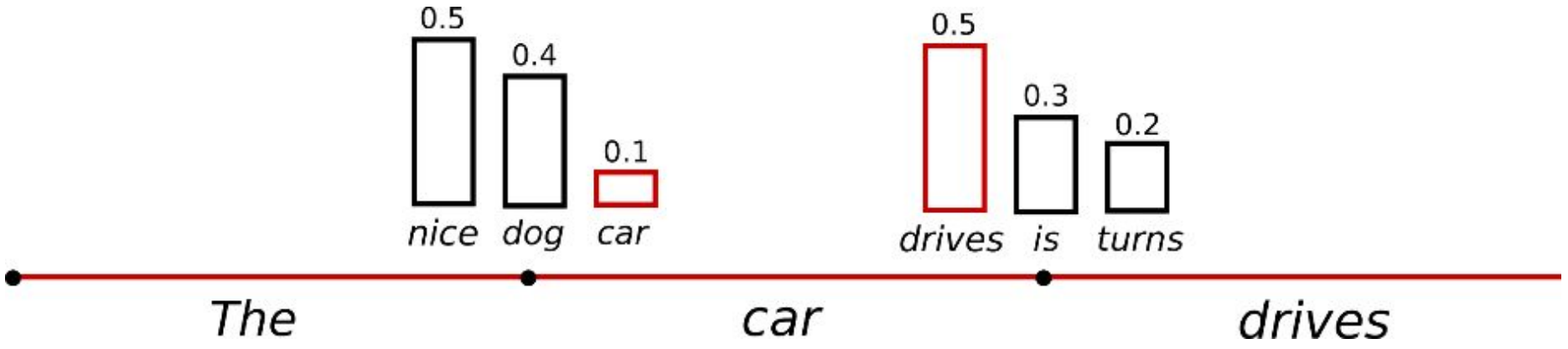
Beam Search

...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...

Decoding as sampling



Ancestral sampling: $p_\theta(y_t|y_{<t}, x)$

Decoding as 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.

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 ..."

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."

Figure 1: Even with substantial human context and the powerful GPT-2 Large language model, Beam Search (size 32) leads to degenerate repetition (highlighted in blue) while pure sampling leads to incoherent gibberish (highlighted in red). When $b \geq 64$, both GPT-2 Large and XL (774M and 1542M parameters, respectively) prefer to stop generating immediately after the given context.

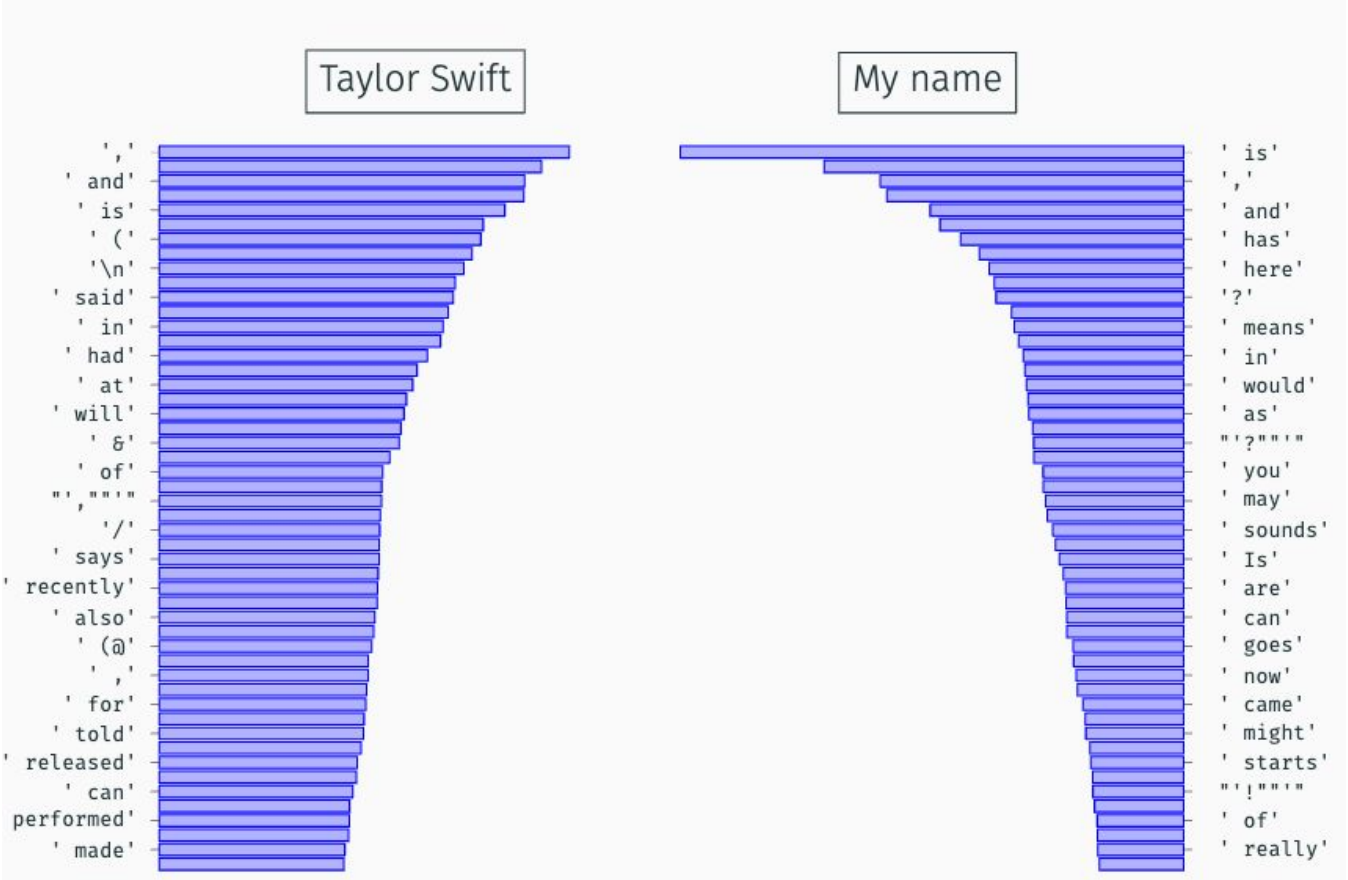
Low-probability tokens are too likely: the distribution has a heavy tail.

Truncation sampling

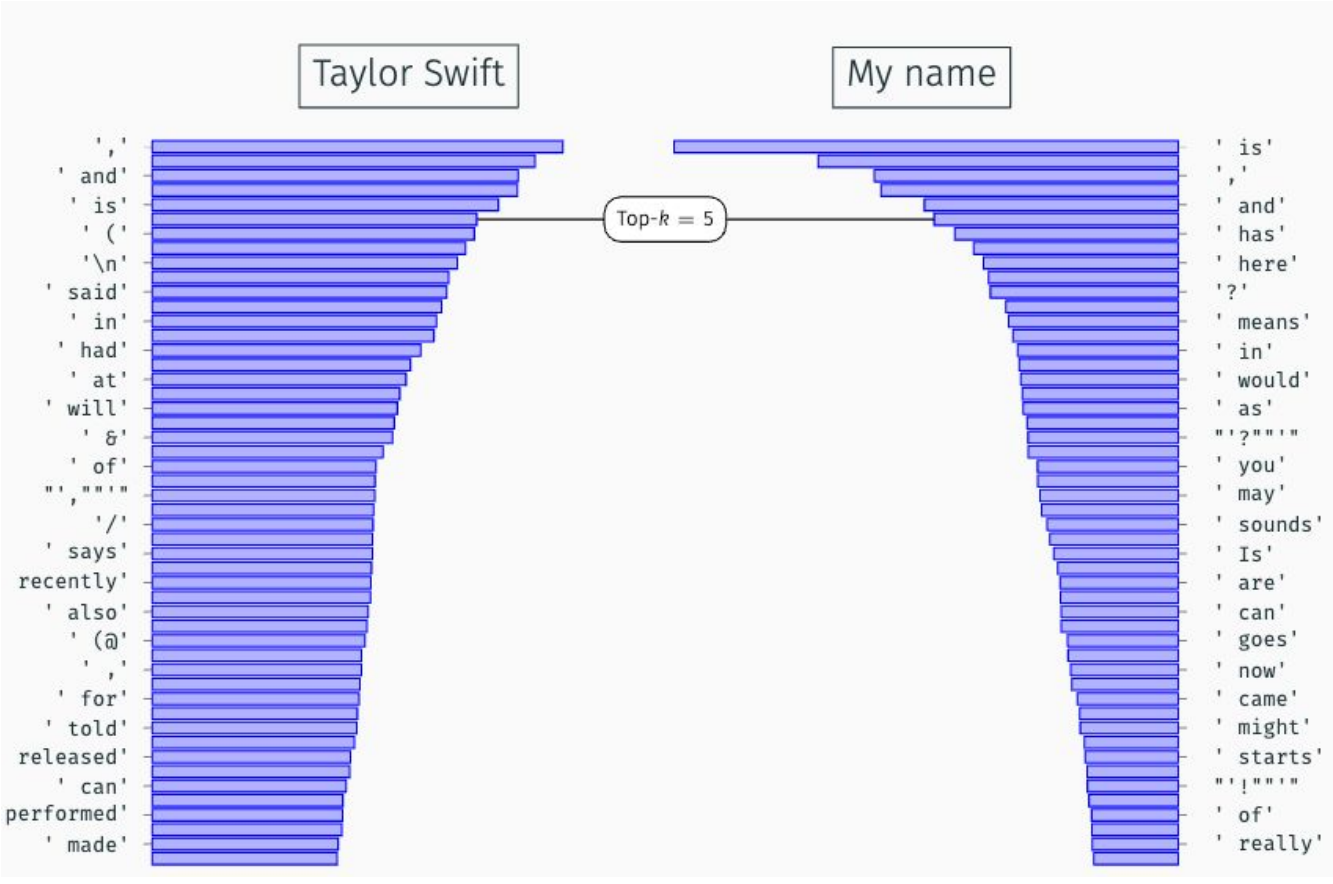
Truncation sampling interpolates greedy and ancestral sampling by choosing a minimum probability threshold at each time step.

- Top-k: sample from k-most probable
- Top-p: Cumulative probability at most p (nucleus sampling)

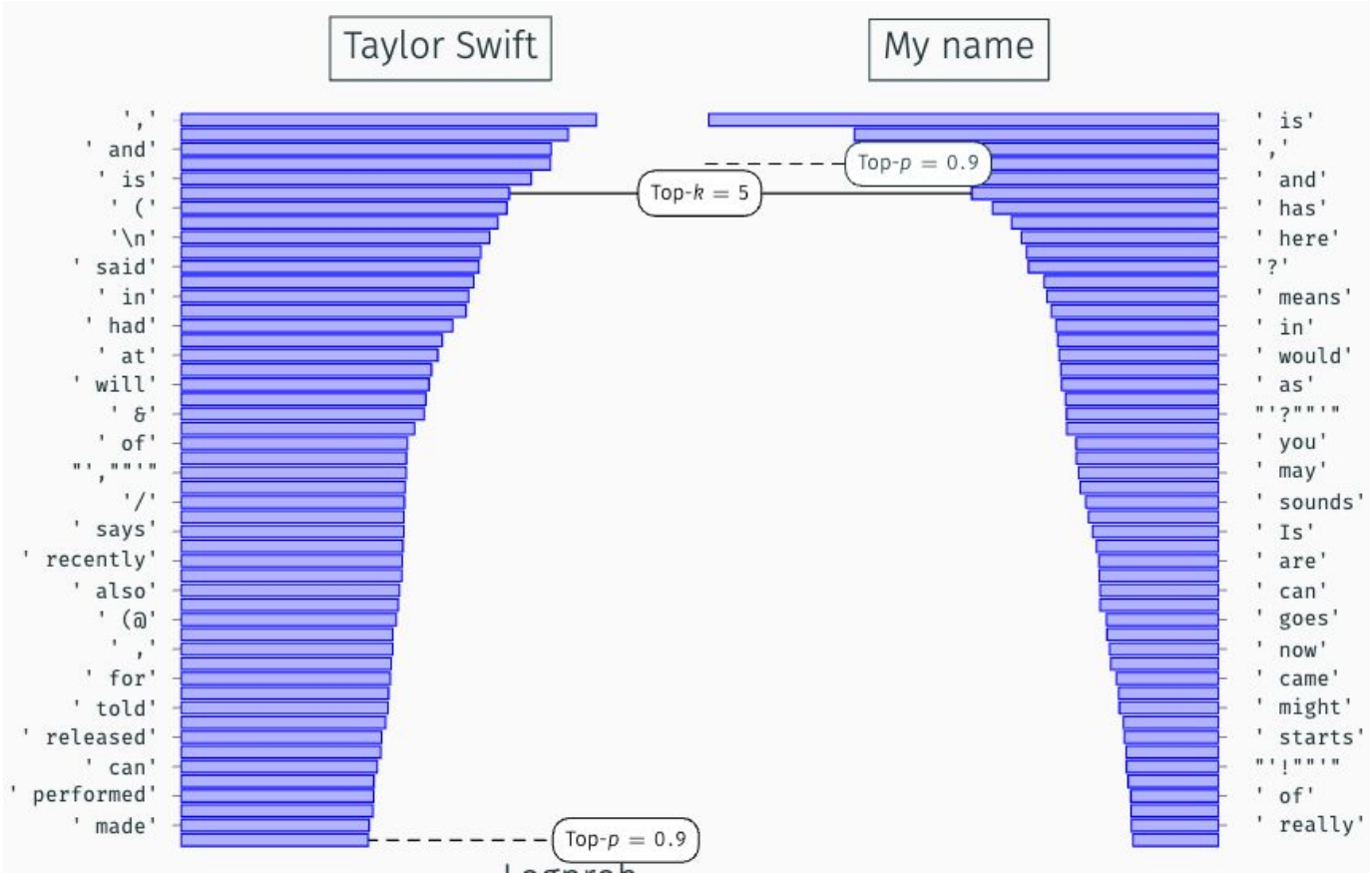
Truncation sampling



Truncation sampling



Truncation sampling



Truncation sampling

```
probs = model(sequence)

# Top-k
topk = probs.topk(k)
indices, weights = topk.indices, topk.values

# Top-p
argsort = probs.argsort(descending=True)
top_p = (argsort.values.cumsum() < p).sum() + 1
indices, weights = argsort.indices[:top_p], argsort.values[:top_p]
```

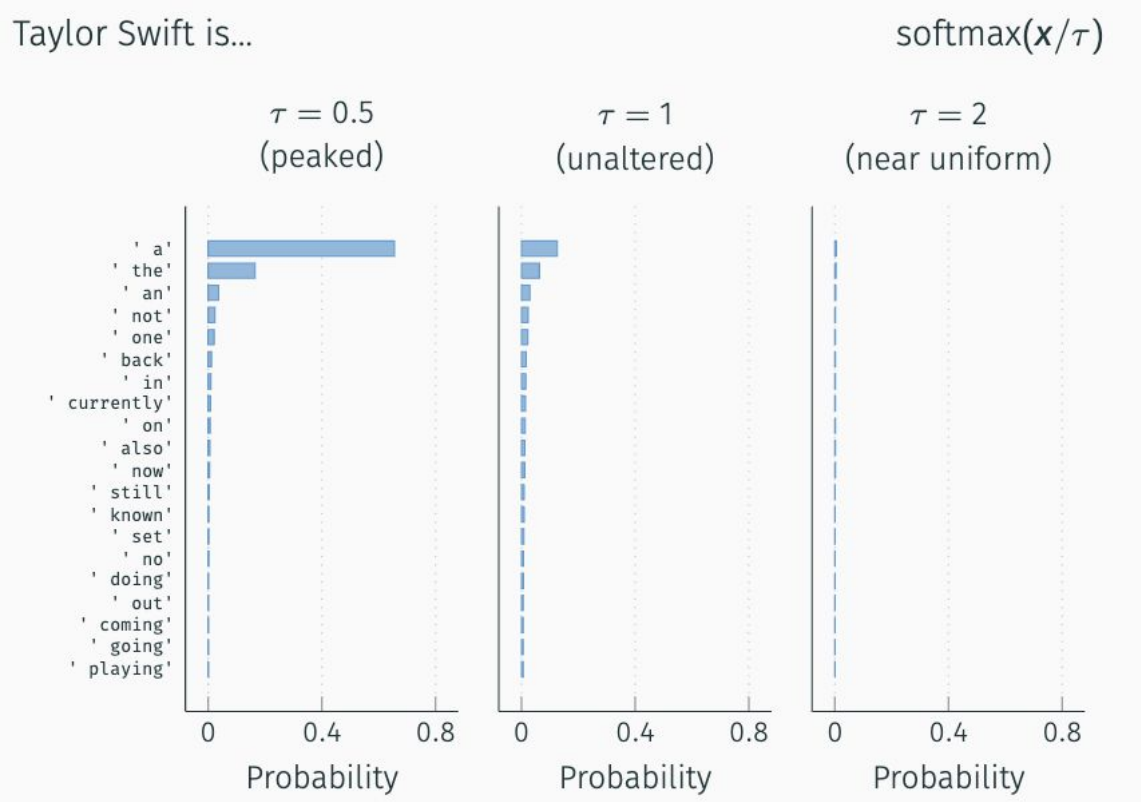
Temperature scaling

Instead of truncating the tail, make the distribution more “peaked”.

$$\text{softmax}(\boldsymbol{x}, \tau) = \frac{\exp(\boldsymbol{x}/\tau)}{\sum_i \exp(x_i/\tau)}$$

Temperature	Parameter	Pro	Con
High	$\tau \geq 1$	Diverse	Incoherent
Low	$\tau < 1$	Coherent	Repetitive

Temperature scaling



nanogpt: temperature scaling with top-k

```python
@torch.no_grad()
def generate(self, idx, max_new_tokens, temperature=1.0, top_k=None):
    """
    Take a conditioning sequence of indices idx (LongTensor of shape (b,t)) and complete
    the sequence max_new_tokens times, feeding the predictions back into the model each time.
    Most likely you'll want to make sure to be in model.eval() mode of operation for this.
    """
    for _ in range(max_new_tokens):
        # if the sequence context is growing too long we must crop it at block_size
        idx_cond = idx if idx.size(1) <= self.config.block_size else idx[:, -self.config.block_size:]
        # forward the model to get the logits for the index in the sequence
        logits, _ = self(idx_cond)
        # pluck the logits at the final step and scale by desired temperature
        logits = logits[:, -1, :] / temperature
        # optionally crop the logits to only the top k options
        if top_k is not None:
            v, _ = torch.topk(logits, min(top_k, logits.size(-1)))
            logits[logits < v[:, [-1]]] = -float('Inf')
        # apply softmax to convert logits to (normalized) probabilities
        probs = F.softmax(logits, dim=-1)
        # sample from the distribution
        idx_next = torch.multinomial(probs, num_samples=1)
        # append sampled index to the running sequence and continue
        idx = torch.cat((idx, idx_next), dim=1)

    return idx
```

WebText

An unprecedented number of mostly young whales have become stranded on the West Australian coast since 2008.

Beam Search, b=16

The number of stranded whales has increased by more than 50 per cent in the past year, with the number of stranded whales on the West Australian coast increasing by more than 50 per cent in the past year. The number of whales stranded on the West Australian coast has increased by more than 50 per cent in the past year, with the number of stranded whales on the West Australian coast increasing by more than 50 per cent in the past year.

Pure Sampling

The Australian Food Safety Authority has warned Australia's beaches may be revitalised this year because healthy seabirds and seals have been on the move. More than 50,000 seabirds, sea mammals and seahorses have been swept into the sea by the Holden CS118 and Adelaide Airport CS300 from 2013. A major white-bat and umidauda migration across Australia is under way in Australia for the first time, with numbers reaching an estimated 50,000.

Sampling, t=0.9

Last week's intense storms and a series of powerful cyclones have been officially blamed for the deaths of at least nine large fin whales near Whitsundays - the largest loss of any species globally. The fin whales: packed in the belly of one killer whale thrashing madly in fear as another tries to bring it to safety. When the colossal animal breached the waters of Whitsundays, he'd been seen tagged for a decade.

Top-k, k=640

Pumping Station #3 shut down due to construction damage Find more at:
www.abc.net.au/environment/species-worry/
in-the-top-10-killer-whale-catastrophes-in-history.html
"In the top 10 killer whale catastrophes in history:
1) 1986: Up to 12 orcas struck by lightning; many drowned and many more badly injured.

Top-k, k=40, t=0.7

The whale's fate was confirmed late last week when the animal was found by fishermen off the coast of Bundaberg. Experts believe the whale was struck by a fishing vessel off the coast of Bundaberg, and died after being sucked into the ocean. The whale's fate was confirmed late last week when the animal was found by fishermen off the coast of Bundaberg.

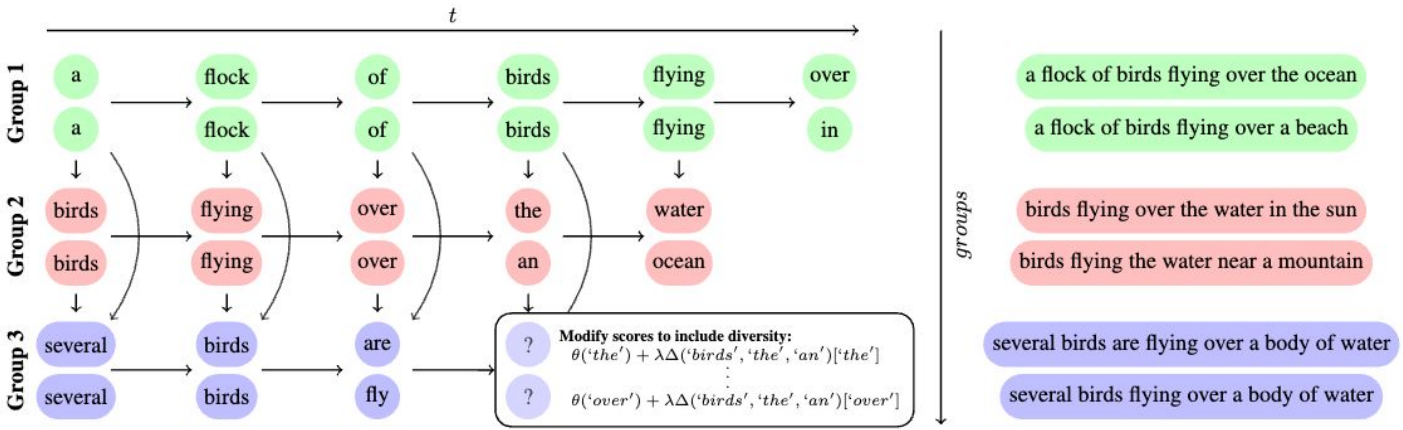
Nucleus, p=0.95

There has been an unprecedented number of calves caught in the nets of whaling stations that operate in WA. Pilot whales continue to migrate to feeding grounds to feed their calves. They are now vulnerable due to the decline of wild populations; they are restricted to one breeding site each year. Image copyright Yoon Bo Kim But, with sharp decline in wild populations the size of the Petrels are shrinking and dwindling population means there will only be room for a few new fowl.

WebText

Poor nutrition has led to a rise in the number of stranded humpback whales on the West Australian coast, veterinary researchers have said. Carly Holyoake, from Murdoch University, at the Australian Veterinary Association's annual conference in Perth on Wednesday, said an unprecedented number of mostly young whales had become stranded on the coast since 2008.

Diverse Beam Search



Divide beam into groups and ensure diversity between groups

Diverse Beam Search

Algorithm 1: Diverse Beam Search

Perform a diverse beam search with G groups using a beam width of B

for $t = 1, \ldots T$ **do**

    // perform one step of beam search for first group without diversity

    $Y^1_{[t]} \leftarrow \text{argmax}_{(\mathbf{y}^1_{1,[t]},\ldots,\mathbf{y}^1_{B',[t]})} \sum_{b\in[B']} \Theta(\mathbf{y}^1_{b,[t]})$

    for $g = 2, \ldots G$ **do**

        // augment log-probabilities with diversity penalty

        $\Theta(\mathbf{y}^g_{b,[t]}) \leftarrow \Theta(\mathbf{y}^g_{b,[t]}) + \sum_h \lambda_g \Delta(\mathbf{y}^g_{b,[t]}, Y^h_{[t]}) \quad b \in [B'], \mathbf{y}^g_{b,[t]} \in \mathcal{Y}^g \text{ and } \lambda_g > 0$

        // perform one step of beam search for the group

        $Y^g_{[t]} \leftarrow \text{argmax}_{(\mathbf{y}^g_{1,[t]},\ldots,\mathbf{y}^g_{B',[t]})} \sum_{b\in[B']} \Theta(\mathbf{y}^g_{b,[t]})$

Return set of B solutions, $Y_{[T]} = \bigcup_{g=1}^{G} Y^g_{[T]}$

Unlikelihood training

The key idea behind unlikelihood training is decreasing the model's probability of certain tokens, called *negative candidates*. Given a sequence $(x_1, \ldots, x_T)$ and a set of negative candidate tokens $\mathcal{C}^t = \{c_1, \ldots, c_m\}$, where each $c_j \in \mathcal{V}$, we define the **unlikelihood loss** for step t as:

$$\mathcal{L}_{\text{UL}}^t(p_\theta(\cdot|x_{<t}), \mathcal{C}^t) = -\sum_{c \in \mathcal{C}^t} \log(1 - p_\theta(c|x_{<t})). \tag{3}$$

The loss decreases as $p_\theta(c|x_{<t})$ decreases. We incorporate the unlikelihood loss into a **token-level unlikelihood objective** which augments each time-step of maximum likelihood training:

$$\mathcal{L}_{\text{UL-token}}^t(p_\theta(\cdot|x_{<t}), \mathcal{C}^t) = -\alpha \cdot \underbrace{\sum_{c \in \mathcal{C}^t} \log(1 - p_\theta(c|x_{<t}))}_{\text{unlikelihood}} - \underbrace{\log p_\theta(x_t|x_{<t})}_{\text{likelihood}}. \tag{4}$$

As candidates, we use previous context tokens:

$$\mathcal{C}_{\text{prev-context}}^t = \{x_1, \ldots, x_{t-1}\} \setminus \{x_t\}. \tag{5}$$

Intuitively, minimizing the unlikelihood loss with this candidate set makes (i) incorrect repeating tokens less likely, as the previous context contains potential repeats, and (ii) frequent tokens less likely, as these tokens appear often in the previous context. These candidates are efficient to compute, without requiring additional supervision.

Contrastive training

Our goal is to encourage the language model to learn discriminative and isotropic token representations. To this end, we introduce a contrastive objective $\mathcal{L}_{\text{CL}}$ into the training of the language model. Specifically, given a variable-length sequence $\boldsymbol{x} = \{x_1, ..., x_{|\boldsymbol{x}|}\}$, the $\mathcal{L}_{\text{CL}}$ is defined as

$$\mathcal{L}_{\text{CL}} = \frac{1}{|\boldsymbol{x}| \times (|\boldsymbol{x}|-1)} \sum_{i=1}^{|\boldsymbol{x}|} \sum_{j=1, j \neq i}^{|\boldsymbol{x}|} \max\{0, \rho - s(h_{x_i}, h_{x_i}) + s(h_{x_i}, h_{x_j})\}, \tag{2}$$

where $\rho \in [-1, 1]$ is a pre-defined margin and h_{x_i} is the representation of token x_i produced by the model. The similarity function s computes the cosine similarity between token representations as

$$s(h_{x_i}, h_{x_j}) = \frac{h_{x_i}^{\top} h_{x_j}}{\|h_{x_i}\| \cdot \|h_{x_j}\|}. \tag{3}$$

Intuitively, by training with $\mathcal{L}_{\text{CL}}$, the model learns to pull away the distances between representations of distinct tokens.[2] Therefore, a discriminative and isotropic model representation space can be obtained. The overall training objective $\mathcal{L}_{\text{SimCTG}}$ is then defined as

$$\mathcal{L}_{\text{SimCTG}} = \mathcal{L}_{\text{MLE}} + \mathcal{L}_{\text{CL}}, \tag{4}$$

where the maximum likelihood estimation (MLE) objective $\mathcal{L}_{\text{MLE}}$ is described in Eq. (1). Note that, when the margin ρ in $\mathcal{L}_{\text{CL}}$ equals to 0, the $\mathcal{L}_{\text{SimCTG}}$ degenerates to the vanilla MLE objective $\mathcal{L}_{\text{MLE}}$.

Contrastive search

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

When generating output, contrastive search jointly considers (i) the probability predicted by the language model to maintain the semantic coherence between the generated text and the prefix text; and (ii) the similarity with respect to the previous context to avoid model degeneration.

Contrastive Framework for Neural Text Generation

Model	Decoding Method	Coherence	Fluency	Informativeness
Agreement	-	0.51	0.64	0.70
MLE	nucleus	2.92	3.32	3.91
	contrastive	2.78	2.29	2.56
Unlikelihood	nucleus	2.59	3.02	3.58
	contrastive	2.76	2.90	3.35
SimCTG	nucleus	2.96	3.34	3.96
	contrastive	3.25★	3.57★	3.96
SimCTG-large	nucleus	3.01	3.37	**3.98**
	contrastive	**3.33**★	**3.66**★	**3.98**
Human	-	3.70	3.71	4.21

Contrastive Framework for Neural Text Generation

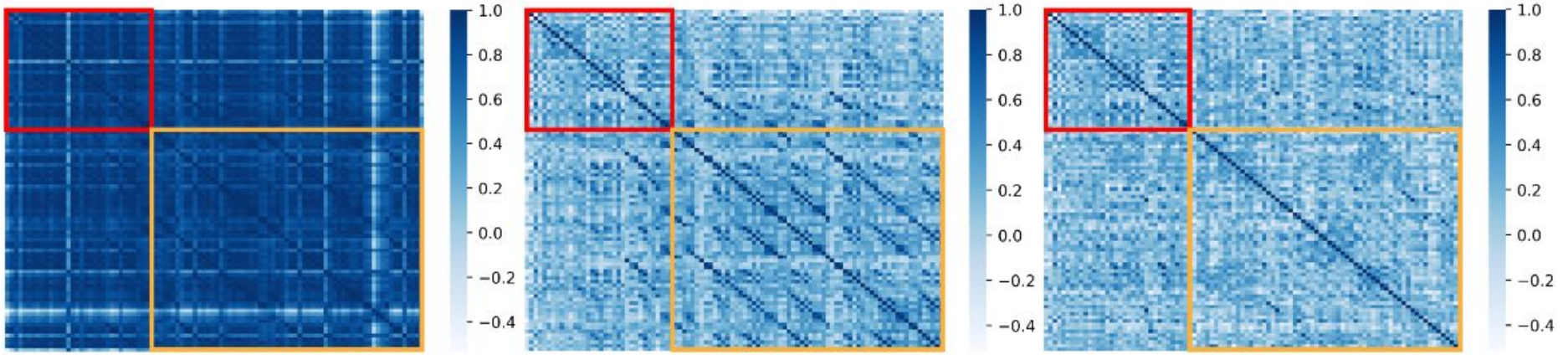


Figure 6: (a) MLE + beam search; (b) SimCTG + beam search; (c) SimCTG + contrastive search. The token similarity matrix of the prefix and the generated text are highlighted in red and yellow.