- conditional text generation is generate text depends on prompt coming from the prompt
- The simplest decoding method to get discrete tokens from a model's continuous out- put is to greedily select the token with the highest probability
- Instead of decoding the token with the highest probability at each step, beam search keeps track of the top-b most probable next tokens, where b is referred to as the num- ber of beams or partial hypotheses. The next set of beams are chosen by considering all possible next-token extensions of the existing set and selecting the b most likely extensions.
- Now let's compare this to a sequence that is generated with beam search. To activate beam search with the generate() function we just need to specify the number of beams with the num_beams parameter.
- The more beams we choose, the better the result potentially gets; however, the generation process becomes much slower since we generate parallel sequences for each beam:
- beam search keep working on b probabilities and get its all sum till it get best probability of generation text
- We can see that we get a better log probability (higher is better) with beam search than we did with simple greedy decoding. However, we can see that beam search also suffers from repetitive text. One way to address this is to impose an n-gram penalty with the no_repeat_ngram_size parameter that tracks which n-grams have been seen and sets the next token probability to zero if it would produce a previously seen n-gram
- temperature act on how model going on creativity in generation text
- Top-k and nucleus (top-p) sampling are two popular alternatives or extensions to using temperature. In both cases, the basic idea is to restrict the number of possible tokens we can sample from at each timestep

Top-k Sampling:

- **Use When:** You want a fixed, manageable number of high-probability options.
- **Scenarios:** Tasks requiring a more predictable and stable output, such as summarization or translation.

Top-p Sampling:

- **Use When:** You need adaptive flexibility to handle varying contexts and ensure a broader range of possibilities.
- **Scenarios:** Creative writing, dialogue generation, or any task where balancing coherence and diversity is crucial.

- The idea behind top-k sampling is to avoid the low-probability choices by only sam- pling from the k tokens with the highest probability. This puts a fixed cut on the long tail of the distribution and ensures that we only sample from likely choices
- The value of k is chosen manually and is the same for each choice in the sequence, independent of the actual output distribution. We can find a good value for k by looking at some text quality metrics
- Unfortunately, there is no universally "best" decoding method. Which approach is best will depend on the nature of the task you are generating text for. If you want your model to perform a precise task like arithmetic or providing an answer to a spe- cific question, then you should lower the temperature or use deterministic methods like greedy search in combination with beam search to guarantee getting the most likely answer. If you want the model to generate longer texts and even be a bit crea- tive, then you should switch to sampling methods and increase the temperature or use a mix of top-k and nucleus sampling.