# Decoding Strategies

# Deterministic Decoding

### Learning goals

- Get to know deterministic decoding strategies
- Learn how text is generated with greedy search and beam search
- Understand how beam search tries to fix the drawbacks of greedy search

# GREEDY SEARCH (1)



- **Core idea**: Greedy search selects the word with the highest probability at each timestep, iteratively building the output sequence
- **Exploration of search space**: It explores a single path through the output space, favoring the most probable word at each step without considering future consequences
- **Candidate Sequence**: Only keeps track of the most likely sequence at each step, discarding other possibilities
- **Decision Making**: It makes local decisions based solely on the highest probability at the current step without considering potential longer-term outcomes

# GREEDY SEARCH (2)

- The model accepts an input sequence of tokens $x_1, x_2, ..., x_N$, which we also call prompt
- The model then generates a token at each timestep $t$ until $T$: $y_1, y_2, ..., y_T$
- In greedy search we choose the token with the highest conditional probability from the vocabulary V
- $y_t = argmax_{y \in V} P(y|y_1, y_2, ..., y_{t-1}, \mathbf{x})$
- With $y_t$ being the chosen token at timestep t and $\mathbf{x} = (x_1, x_2, ..., x_N)$ being the initial prompt

# GREEDY SEARCH: EXAMPLE

▸ d2l book

- Suppose our vocabulary only has four tokens: *A*, *B*, *C* and <eos>

| Time step | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| A | 0.5 | 0.1 | 0.2 | 0.0 |
| B | 0.2 | 0.4 | 0.2 | 0.2 |
| C | 0.2 | 0.3 | 0.4 | 0.2 |
| <eos> | 0.1 | 0.2 | 0.2 | 0.6 |

- At each timestep greedy search chooses the token with the highest conditional probability
- The model thus predicts *A*, *B*, *C*, <eos>
- Its probability is $0.5 \cdot 0.4 \cdot 0.4 \cdot 0.6 = 0.048$

# DRAWBACKS OF GREEDY SEARCH (1)

- Now we select token *C* at timestep 2 instead of *B*

| Time step | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| A | 0.5 | 0.1 | 0.1 | 0.1 |
| B | 0.2 | 0.4 | 0.6 | 0.2 |
| C | 0.2 | 0.3 | 0.2 | 0.1 |
| <eos> | 0.1 | 0.2 | 0.1 | 0.6 |

- At the timesteps 3 and 4 the conditional probabilities change since the context is no longer *A*, *B* but *A*, *C*
- The final token sequence is *A*, *C*, *B*, <eos>
- Its probability is $0.5 \cdot 0.3 \cdot 0.6 \cdot 0.6 = 0.054$
- Even though *C* at $t = 2$ has a lower probability, the final sequence has a higher probability

# DRAWBACKS OF GREEDY SEARCH (2)

- **Suboptimal Global Solutions**: It makes decisions based only on the highest probability token at each step, often missing globally optimal solutions
- **Lack of Diversity**: It generates repetitive and predictable text, leading to bland outputs
- **Incoherence in Long Sequences**: It may produce incoherent text over longer sequences due to losing track of the overall context
- **Repetitiveness**: The lack of diversity leads to repetitive phrases, especially in longer texts
- **Overemphasis on Common Phrases**: It favors common words and phrases, resulting in overly generic outputs

# BEAM SEARCH

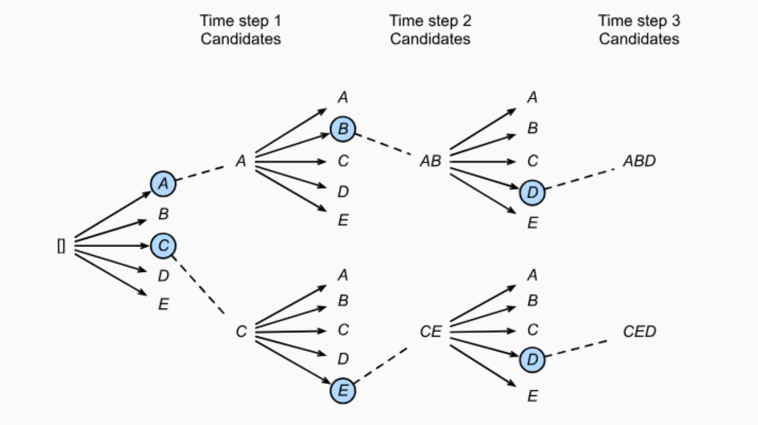


- **Core idea**: Beam search extends the exploration to multiple possible sequences instead of just the most probable one
- **Exploration of search space**: It explores multiple paths (or "beams") simultaneously, maintaining a set of promising candidate sequences
- **Candidate Sequence**: Keeps a fixed number of most probable sequences (determined by the beam width parameter $k$) at each step
- **Decision Making**: At each step, it considers multiple candidate sequences and selects the most probable ones based on their cumulative probabilities up to that point

# BEAM SEARCH: EXAMPLE (1)

▸ d2l book

- Suppose $V = \{A, B, C, D, E\}$ and beam width $k = 2$



# BEAM SEARCH: EXAMPLE (2)

- At each timestep beam search chooses the $k$ tokens with the highest joint probability
- Suppose at $t = 1$ $A$ and $B$ have the highest conditional probabilities $P(y_1|\mathbf{x})$
- At $t = 2$ for all $y_2 \in V$ we compute:

$$P(A, y_2|\mathbf{x}) = P(A|\mathbf{x}) \cdot P(y_2|A, \mathbf{x})$$
$$P(C, y_2|\mathbf{x}) = P(C|\mathbf{x}) \cdot P(y_2|C, \mathbf{x})$$

- And we again pick the $k$ sequences with the highest probabilities ($AB$ and $CE$)
- At $t = 3$ again for all $y_3 \in V$ we compute:

$$P(A, B, y_2|\mathbf{x}) = P(A, B|\mathbf{x}) \cdot P(y_2|A, B, \mathbf{x})$$
$$P(C, E, y_2|\mathbf{x}) = P(C, E|\mathbf{x}) \cdot P(y_2|C, E, \mathbf{x})$$

- We repeat this process unitl the maximum length is reached or unitl the <EOS> token gets generated

# BEAM WIDTH: PROS AND CONS

**Advantages**:

- **Better Quality**: More likely to find a globally optimal sequence, producing higher quality and more coherent text
- **Balance Between Exploration and Exploitation**: Avoids the pitfalls of greedy search
- **Flexibility**: Beam width can be adjusted to trade off between computational complexity and output quality

**Drawbacks**:

- **Computational Complexity**: More computationally intensive than greedy search
- **Limited Diversity**: May still produce similar sequences if the beam width is not large enough
- **Hyperparameter Tuning**: Additionally requires to tune the hyperparameter $k$