Decoding Strategies

What is Decoding?

Learning goals

- Get to know the concept of decoding in NLP
- Learn about different decoding strategies

REMINDER: ARLM

- In Autoregressive Language Modeling (ARLM) the model predicts the next token given the previous tokens
- Given the context a language model produces a probability distribution over all the tokens in the vocabulary
- The context is the prompt given to the model plus the already generated tokens
- The way we then choose the next token from that probability distribution to generate natural text is called a decoding strategy

DECODING EXAMPLE (1)

Prompt: Once upon a time

Time step 1:

- Model input: Once upon a time
- Next token: there

Time step 2:

- Model input: Once upon a time there
- Next token: was

Time step 3:

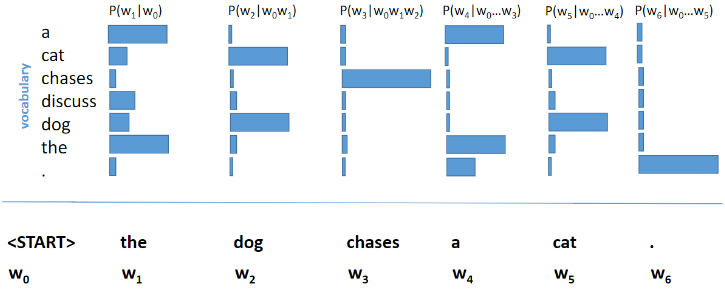
- Model input: Once upon a time there was
- Next token: a

Time step 4:

- Model input: Once upon a time there was a
- Next token: cat

...

DECODING EXAMPLE (2)



- At each timestep the model produces a probability distribution
- A decoding strategy determines how to choose the next token from that distribution, that token is then added to the context
- Generation stops based on stopping criteria (*see: next slide*)

STOPPING CRITERIA FOR TEXT GENERATIONS

- **<EOS> Token**: When this token is generated the model stops
- **Maximum Length**: A predefined maximum length can be set for the generated text. When the text reaches this length, generation stops to prevent excessively long outputs
- **Maximum Time**: A predefined maximum time for generation can be set. After this time has been reached, generation stops
- **Other Criteria**: There are more stopping criteria implemented in huggingface ▸ huggingface

GENERATE FUNCTION

There is two types of hyperparameters for the `generate()` function of the `Transformers` library: One that control the length of the ouput and one that control the generation strategy used: ▸ huggingface, GenerationConfig

- You input a tokenized sentence as the context into the `generate()` function
- And then control the length of the output with the following hyperparameters:
    - `max_length`: The maximum length the generated tokens can have. Corresponds to the length of the input prompt + `max_new_tokens`. Its effect is overridden by `max_new_tokens`, if also set
    - `max_new_tokens`: The maximum numbers of tokens to generate, ignoring the number of tokens in the prompt

GENERATE FUNCTION

  - `min_length`: The minimum length of the sequence to be generated. Corresponds to the length of the input prompt + `min_new_tokens`. Its effect is overridden by `min_new_tokens`, if also set
  - `min_new_tokens`: The minimum numbers of tokens to generate, ignoring the number of tokens in the prompt
  - `max_time`: The maximum amount of time you allow the computation to run for in seconds. generation will still finish the current pass after allocated time has been passed
- With the second type of hyperparameters you control which of the following (*see: next slide*) decoding strategies you use (*will be introduced in the following chapters*)

DECODING STRATEGIES

Deterministic

- Greedy search
- Beam search
- Contrastive search ▸ Su et al., 2022
- Contrastive decoding ▸ Li et al., 2023

Stochastic

- Sampling (with temperature)
- Top-*k* sampling ▸ Fan et al., 2018
- Nucleus top-*p* sampling ▸ Holtzman et al., 2019
- Typical sampling ▸ Meister et al., 2023

Remark: Other decoding strategies exist, and various combinations are possible, such as top-*k* sampling with temperature, or top-*p* sampling followed by top-*k* sampling (with temperature), etc.