O'REILLY®

Hands-on Prompt Engineering

Poll

Who has used prompting and generative for work tasks? ChatGPT, Gemini, any other model?

- Use it all the time
- Use it sometimes
- Have never used it for work
- Have never user any generative AI chatbot

Poll

What kind of tasks do you usually perform with generative AI?

- Email and content writing
- Programming and debugging
- Research into topics
- Personal use such as planning meals, trips, parties

Prerequisites

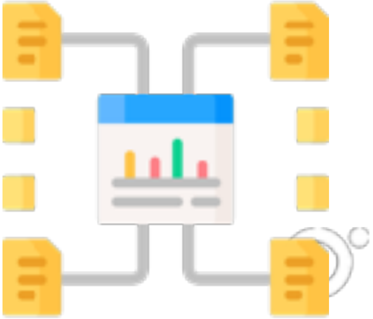
- No prerequisites at all
- Course content geared towards a technical audience
    - Developers, testers, analysts, program and project managers - anyone who works in technology

Prompt Engineering

Prompt engineering is the process of designing and refining prompts to effectively guide the output of language models and other AI systems.

Prompt Engineering

- Natural language text describing the task you want the model to perform
- Primarily used in communication with a text-to-text model
- Enables by in-context learning – ability of model to learn from prompts
- For text-to-image or text-to-audio models defines the kind of output desired



Importance of Prompting

- Guide the model to generate relevant output
- Improve quality and diversity of generated output
- Increase control and interpretability, reduces bias
- Mitigate **hallucination** by guiding the model
- Determine good and bad outcomes by goal setting

Challenges in Prompting

- Get the required results on the first try
- Figuring out the right place to start
- Mitigating bias in the output
- Obtaining diversity and creativity in the results
- Maintain balance between precision and creativity

Where Can We Use Prompt Engineering?

ChatGPT - AI chatbot that took the world by storm

Gemini - Google’s conversational AI service

Claude - Anthropic’s LLM and chatbot

Microsoft Copilot

Llama - Meta’s open source, free LLM and chatbot

Grok - xAI’s chatbot

O'REILLY

Understanding Generative AI and LLMs

Generative AI

Generative AI refers to artificial intelligence systems that can create new content, such as text, images, music, or videos, by learning patterns from existing data.

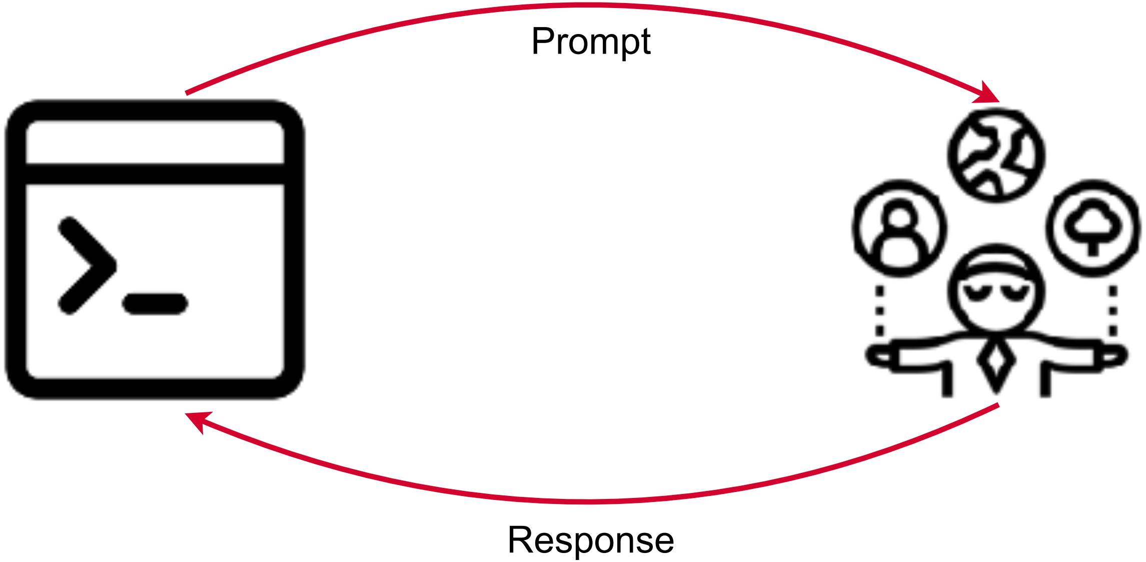
How Do Generative AI Models Work?

- Uses a powerful ML model to learn patterns and relationships in a dataset created by humans
- The model uses learned data to create new content
- The idea is that the new content resembles the content that the model has already seen before

Large Language Models

These are models that process natural language inputs and **predicts the next word based** on what has come before

Model Responses to Prompts



Prompt

Response

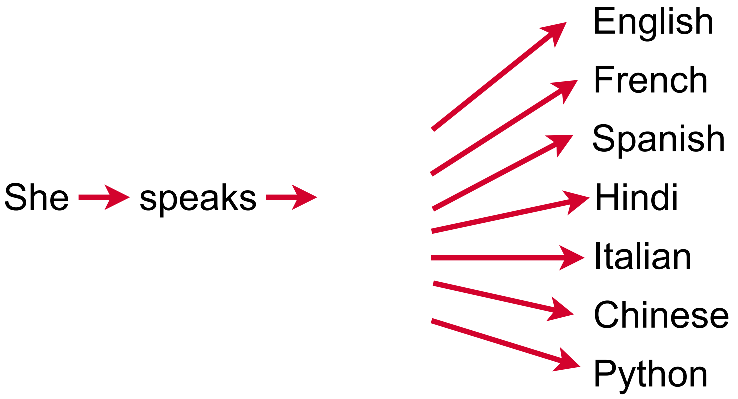
Response a Sequence of Words

She speaks French quite well

Model Generates One Word at a Time

She → speaks → French → quite → well

Many Possible Words at Each Step


She → speaks →

English

French

Spanish

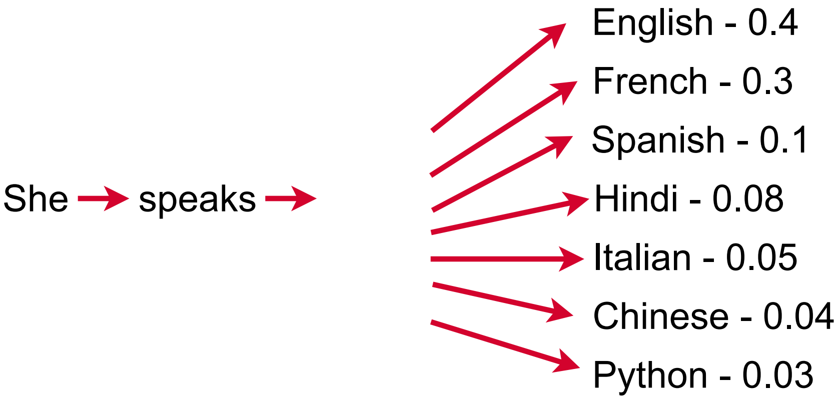
Hindi

Italian

Chinese

Python

Each Possible Next Word is Assigned a Probability



She → speaks →

English - 0.4

French - 0.3

Spanish - 0.1

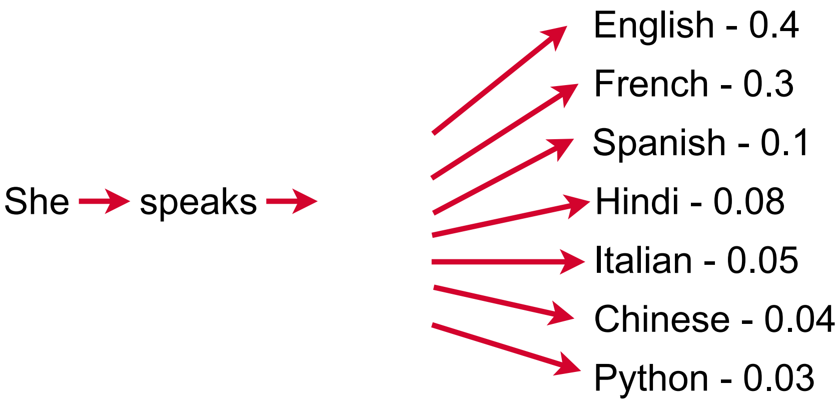
Hindi - 0.08

Italian - 0.05

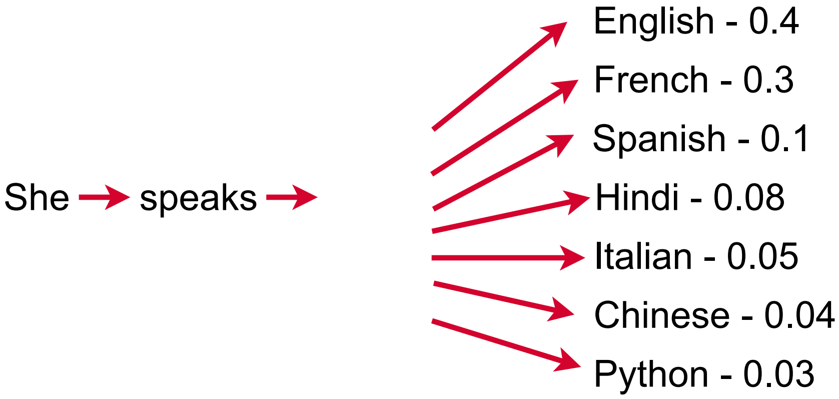
Chinese - 0.04

Python - 0.03

The Model Picks One Word from Possible Next Words



Higher Probability Words are More Likely to be Picked



Response Generated by Picking Words at Each Step

She → speaks → French → quite → well

Large Language Models

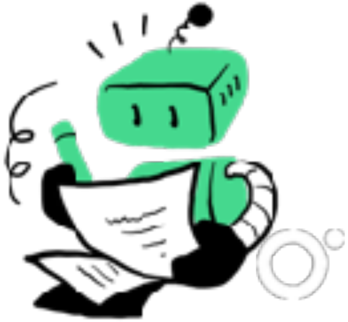
Huge models

Large datasets

What is GPT?

ChatGPT and GPT

- ChatGPT is referred to as a model – but it is the interface to the underlying model that powers it
- ChatGPT is a chatbot built using a text-to-text model called GPT (**G**enerative **P**re-trained **T**ransformer)



Generative Pre-trained Transformer

Generative: Refers to the model's ability to generate text

Pre-trained: Model pre-trained before fine-tuned for specific tasks

Transformer: Type of neural network architecture used by GPT

GPT

Generative Pre-trained Transformers are a type of large language model (LLM) and a prominent framework for generative artificial intelligence.

Other Models

Meta’s **Llama** - another generative transformer-based foundational LLM

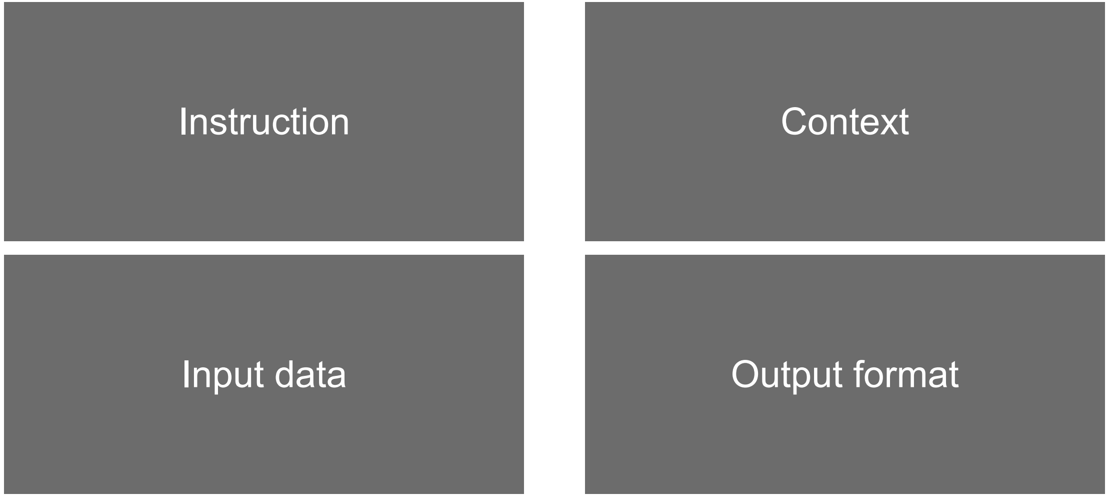
There are other foundational models beyond the GPT series - Google’s **Gemini** or **PaLM**

Anthropic’s **Claude**, another GPT-based model

O'REILLY

Crafting Prompts

Anatomy of a Prompt



Instruction

Context

Input data

Output format

Advanced Techniques in Prompting

- Zero-shot prompting
- Few-shot prompting
- Chain-of-Thought prompting
- Augmented knowledge prompting

Three Steps in Prompt Engineering

Start with a reasonable prompt

Refine, iterate, evaluate, repeat

Calibrate and fine-tune

Start with a Reasonable Prompt

Be precise and clear

Assign roles or personas

Use constraints

Avoid leading or biasing the model

Refine, Iterate, Evaluate, Repeat

- Start somewhere with an initial draft
- Generate and test the response
- Evaluate if the prompt aligns with the objective
- Refine the prompt to guide model in the right direction

Calibrate and Fine-tune

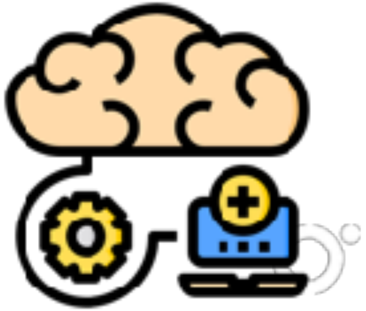
- Advanced techniques to better align the model for specific tasks
- Involves adjusting the model parameters to achieve this

Best Practices for Prompt Design

- Make sure you are using the latest model e.g. ChatGPT uses GPT5
- Limited queries on the free version
- Put instructions at the beginning and separate instructions from text using ###
- Be specific, descriptive, and detailed about context, outcome, length, format, style
- Provide examples for what you want the output to look like

Best Practices for Prompt Design

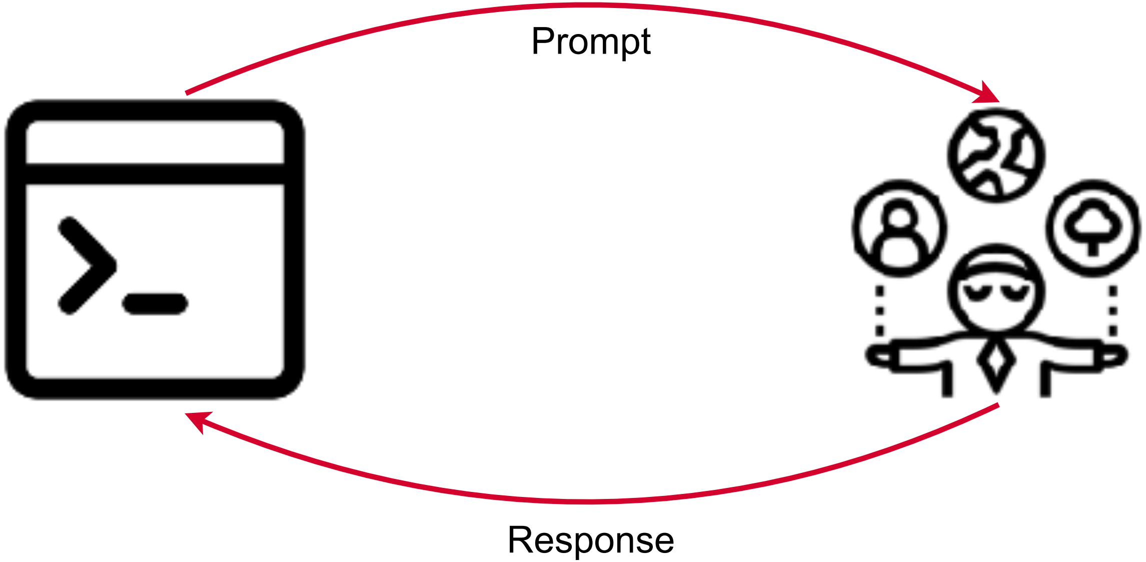
- Start with zero-shot and then use few-shot, and then fine-tune model
- Make descriptions crisp, clear, and unambiguous, and avoid vague language
- Specify ”what to do” rather than “what not to do”
- For code generation, use leading words to guide the model in the right direction



O'REILLY

Text Generation by LLMs

Model Responses to Prompts



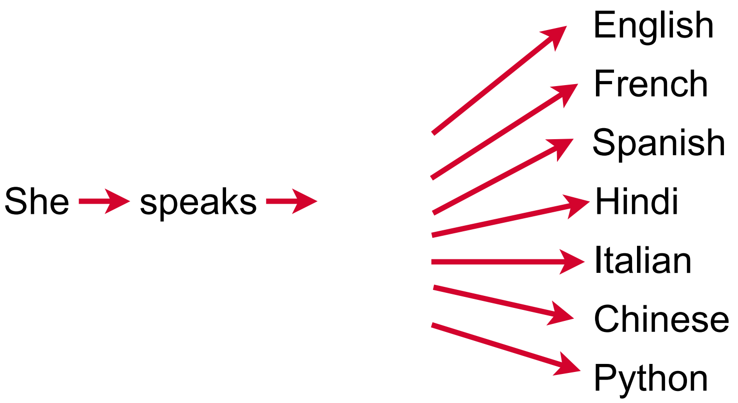
Response a Sequence of Words

She speaks French quite well

Model Generates One Word at a Time

She → speaks → French → quite → well

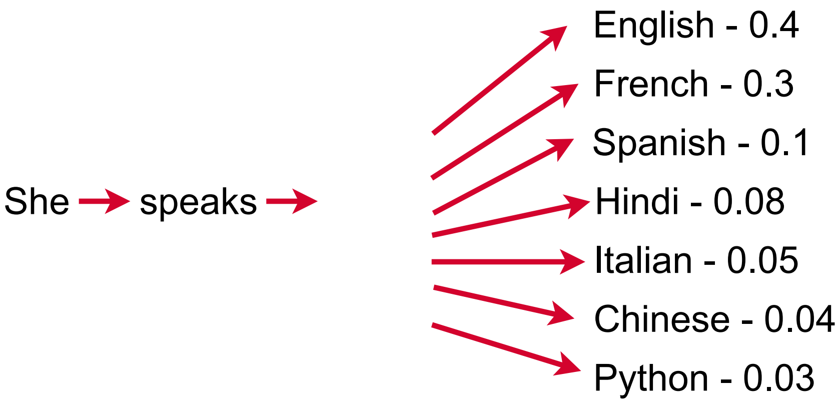
Many Possible Words at Each Step



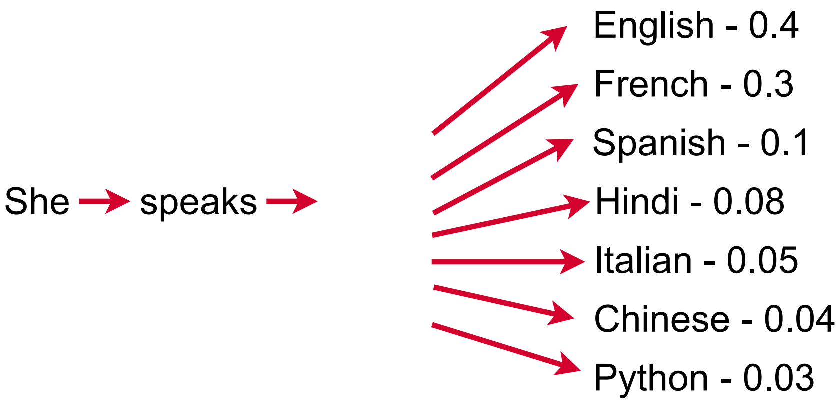
She → speaks →

English
French
Spanish
Hindi
Italian
Chinese
Python

Each Possible Next Word is Assigned a Probability



The Model Picks One Word from Possible Next Words



She → speaks →

English - 0.4

French - 0.3

Spanish - 0.1

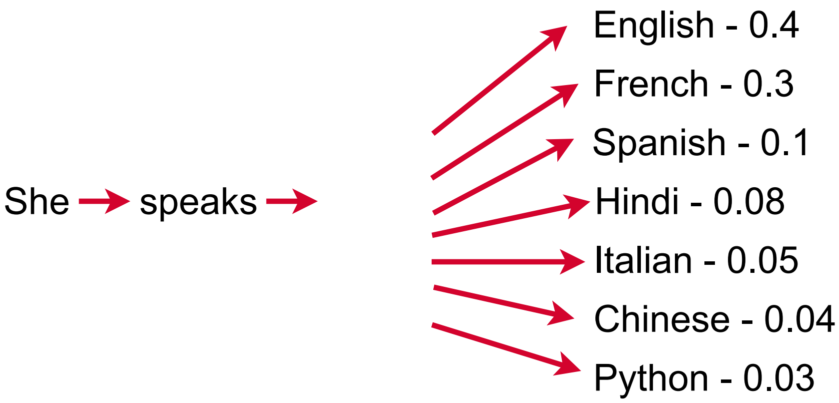
Hindi - 0.08

Italian - 0.05

Chinese - 0.04

Python - 0.03

Higher Probability Words are More Likely to be Picked



She → speaks →

English - 0.4

French - 0.3

Spanish - 0.1

Hindi - 0.08

Italian - 0.05

Chinese - 0.04

Python - 0.03

Response Generated by Picking Words at Each Step

She → speaks → French → quite → well

Model Settings for Tweaking Generated Text

Large language models offer settings that you can tweak to make the generated text **more creative and diverse or more predictable and deterministic**

Creativity vs. Predictability in Text Generation

- High creativity will produce more diverse and unexpected results making the text more engaging
- High predictability generates more consistent and reliable text – useful when you need precise responses
- Striking a balance can produce text that is both interesting and coherent

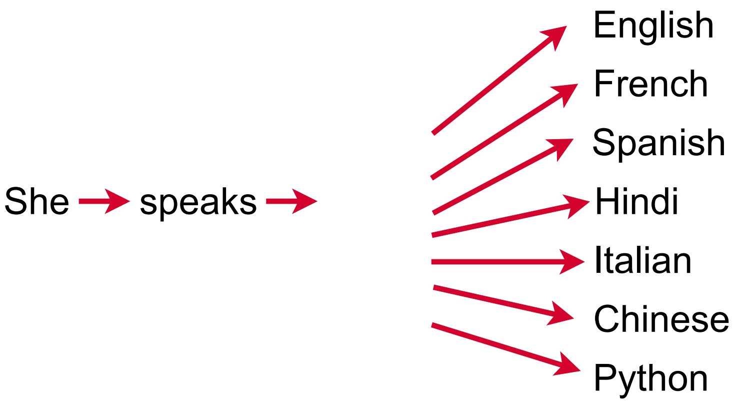
Model Settings to Control Creativity and Predictability

- Temperature
- Top-p (Nucleus Sampling)
- Top-k

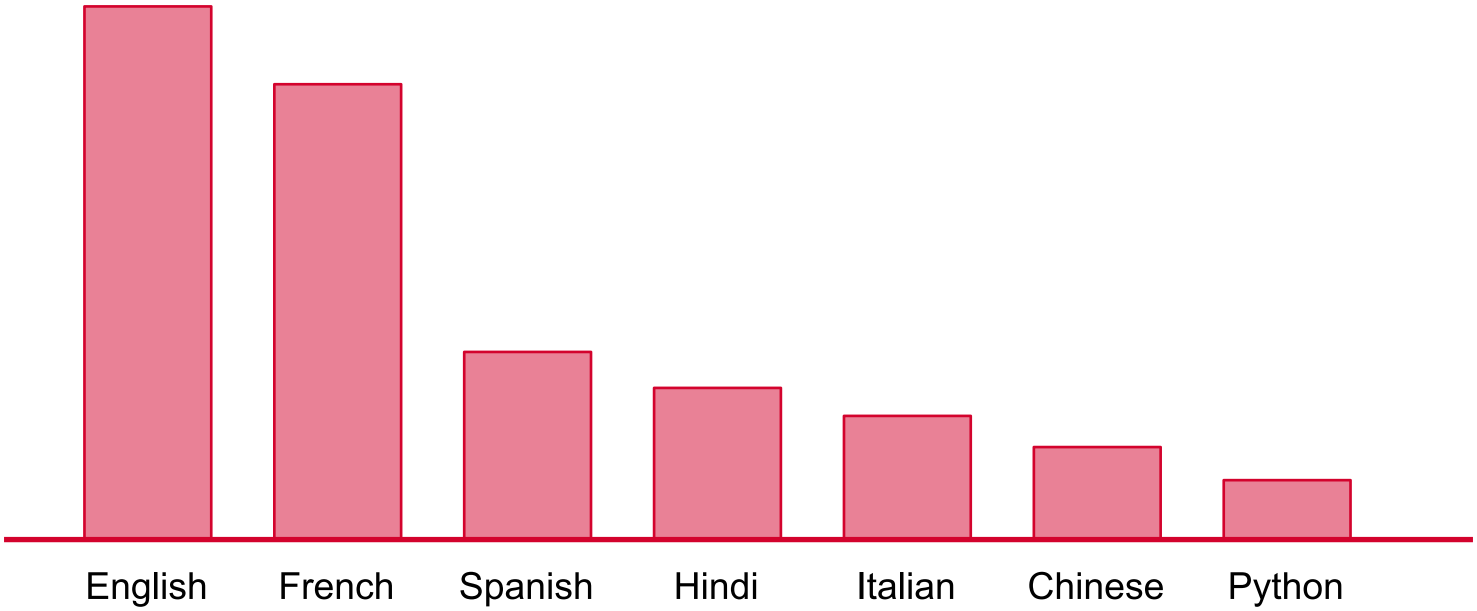
Temperature

- Values range between 0 and 1 (both inclusive)
- Higher values closer to 1 results in more creative output
- Lower values closer to 0 results in more predictable output

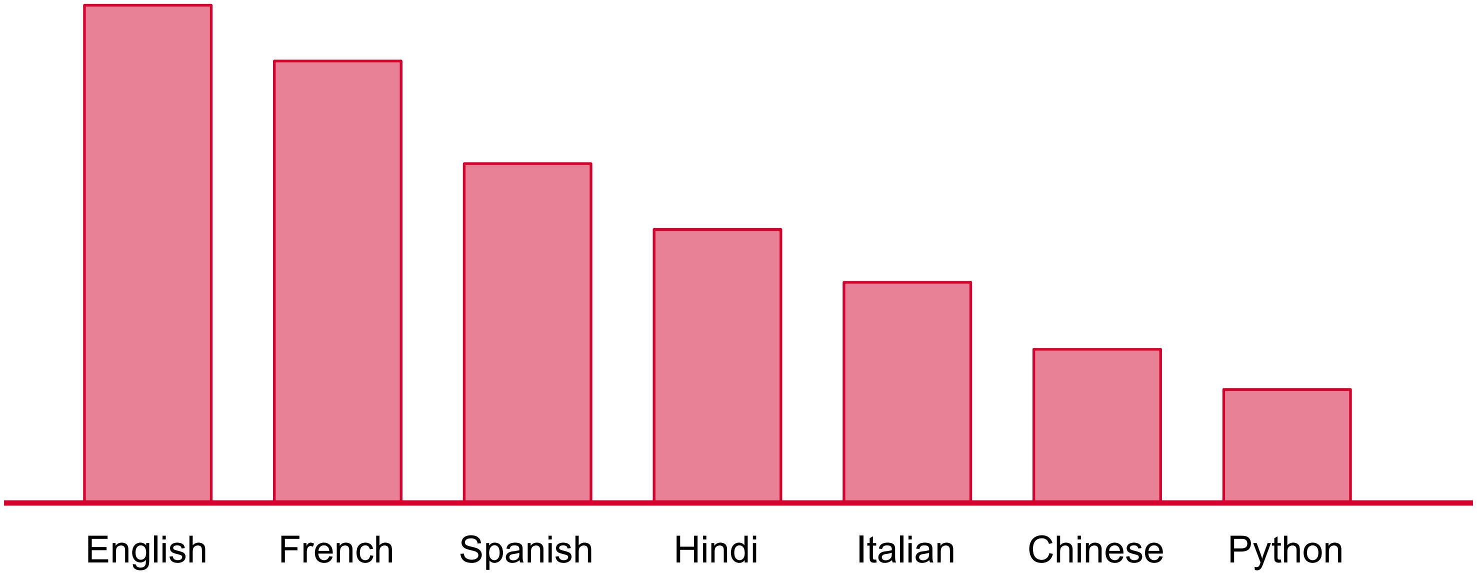
The Model Picks One Word from Possible Next Words



Original Probabilities of Possible Next Words

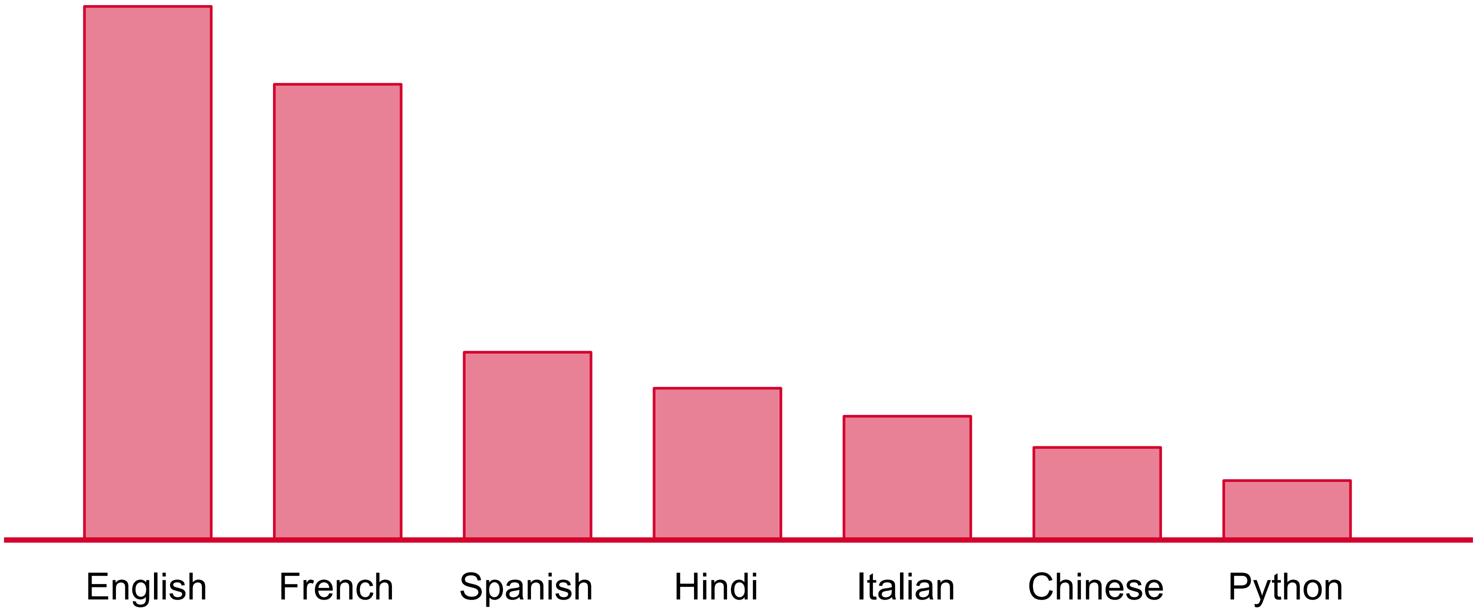


Higher Values of Temperature (Closer to 1)

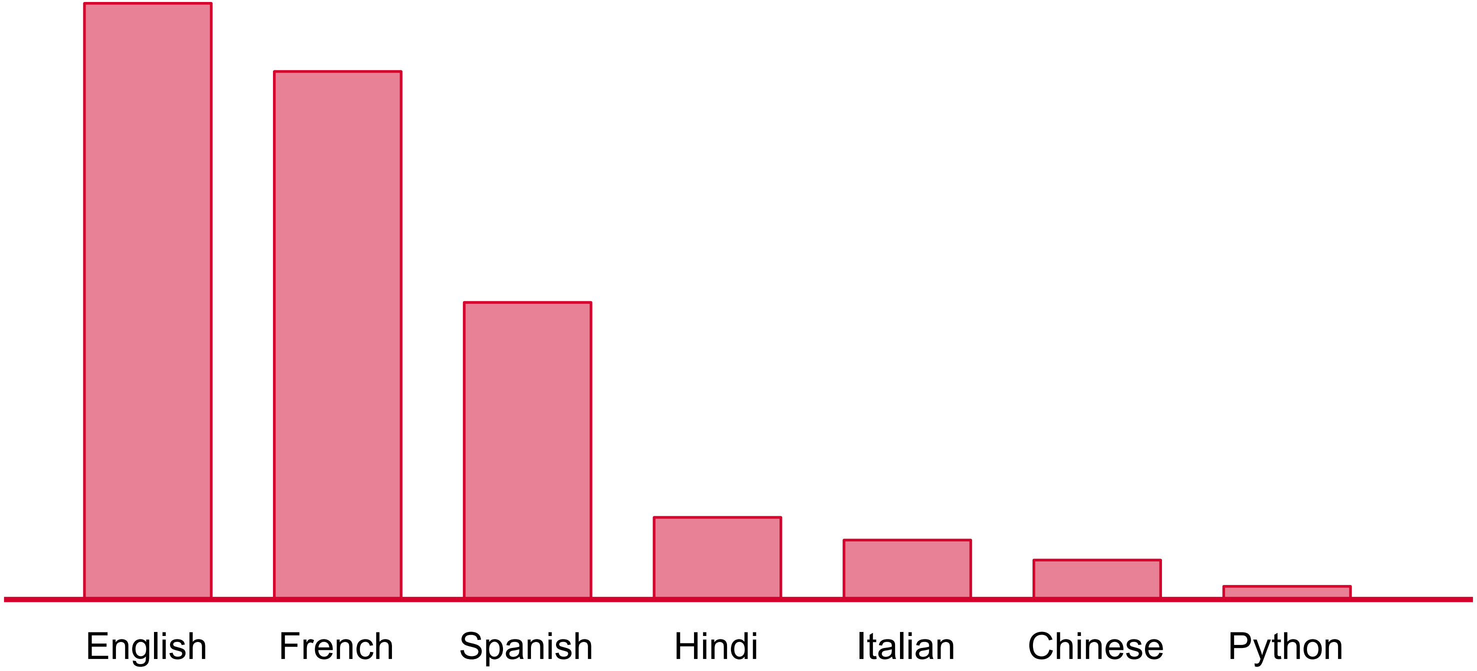


The probability distribution over the next possible word becomes **flatter.**

Original Probabilities of Possible Next Words



Lower Values of Temperature (Closer to 0)



The probability distribution over the next possible word becomes **sharper.**

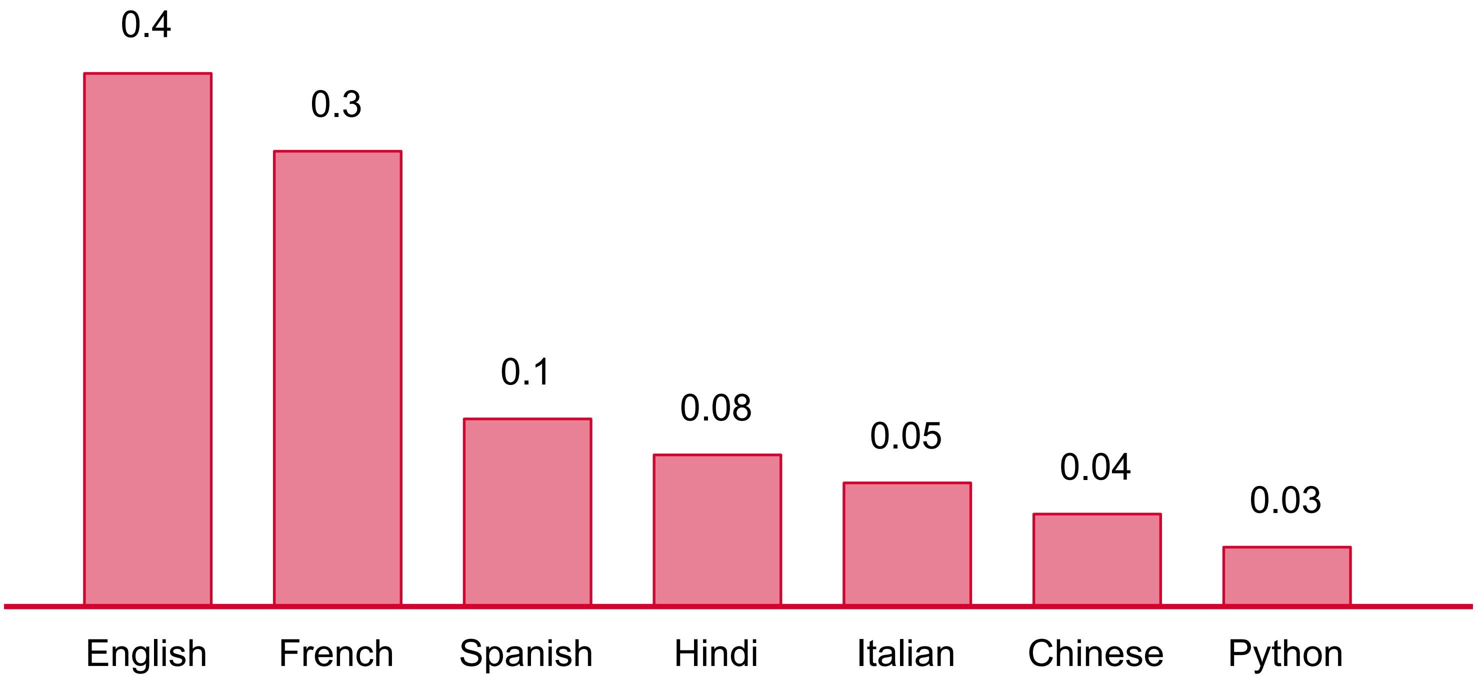
Top-p (Nucleus Sampling)

- Values range between 0 and 1 (both inclusive)
- Values close to 1 result in more diverse and creative output
- Values close to 0 result in more predictable output

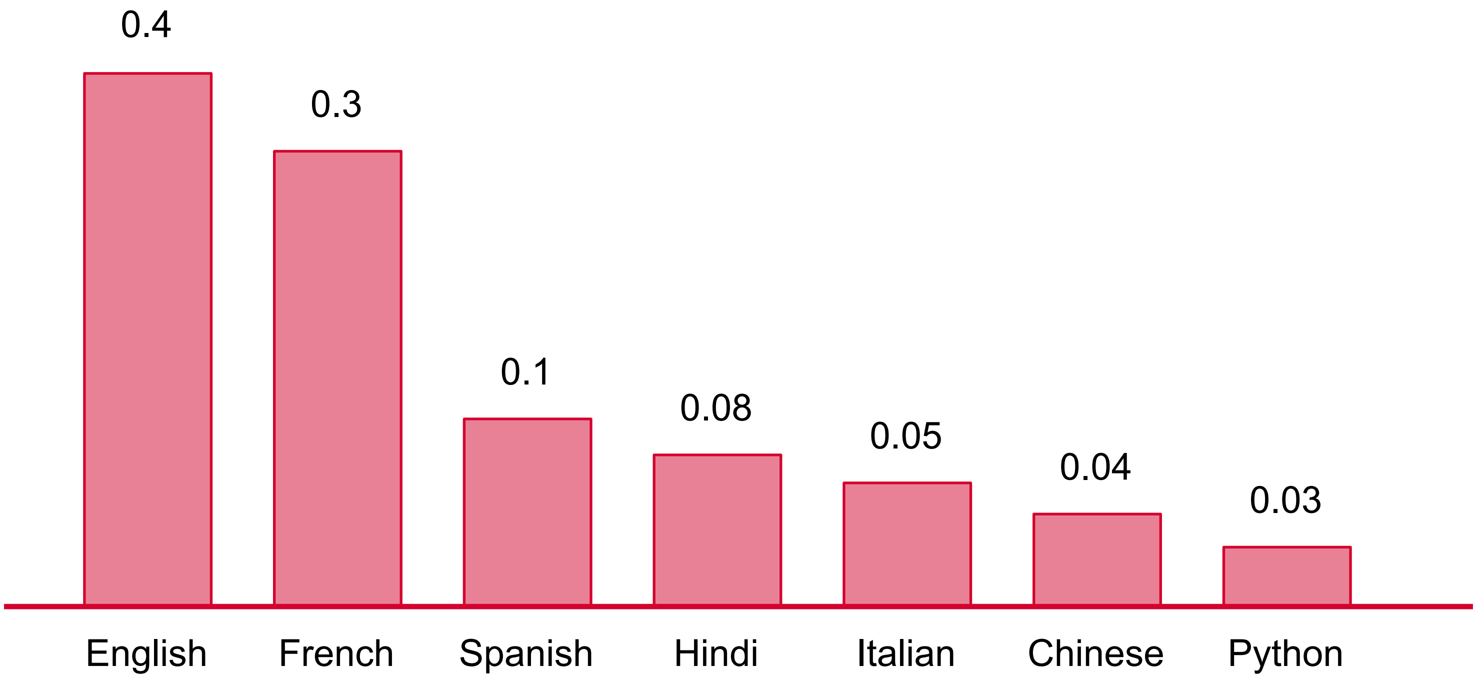
Top-p (Nucleus Sampling)

Top-p sampling, also known as nucleus sampling, works by selecting the **smallest set of top candidate words whose cumulative probability exceeds a given threshold p.**

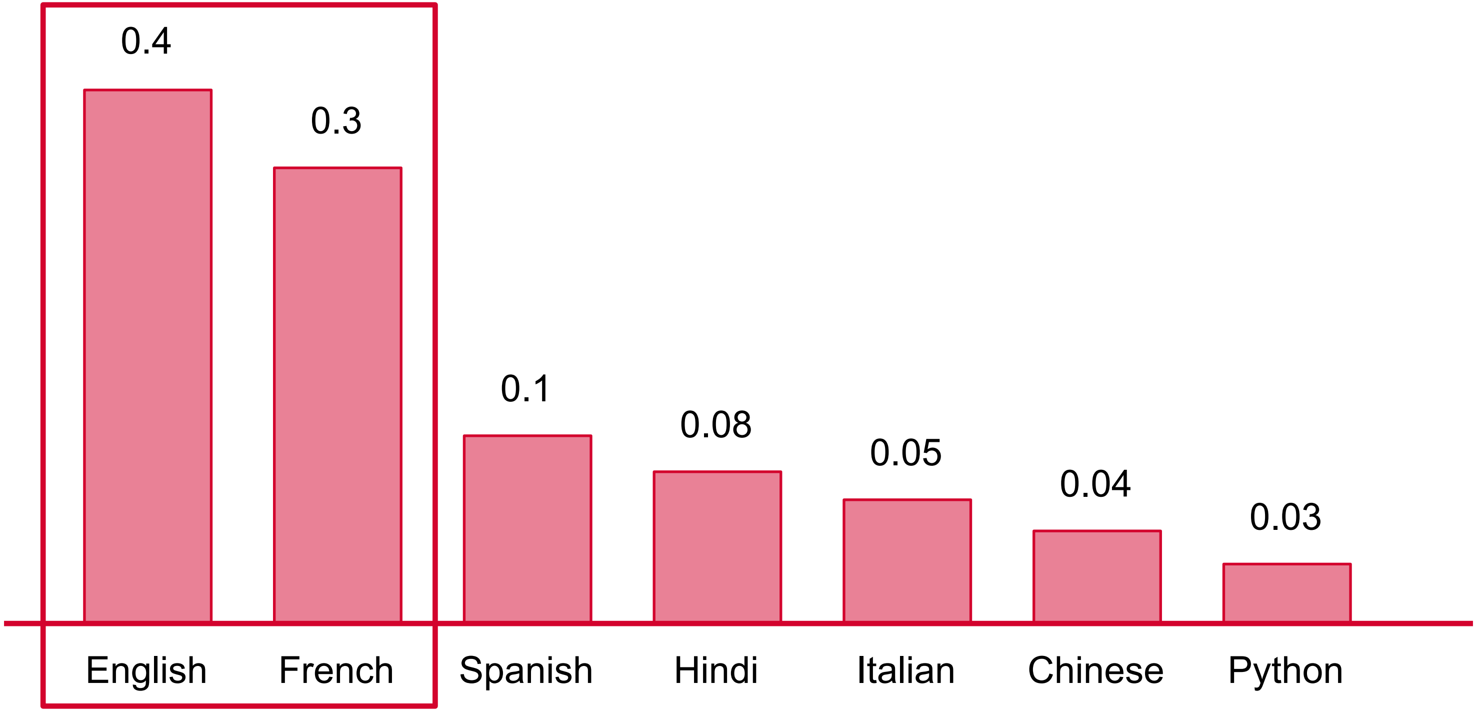
Probabilities of Possible Next Words



Sort All Words in Order of Probability

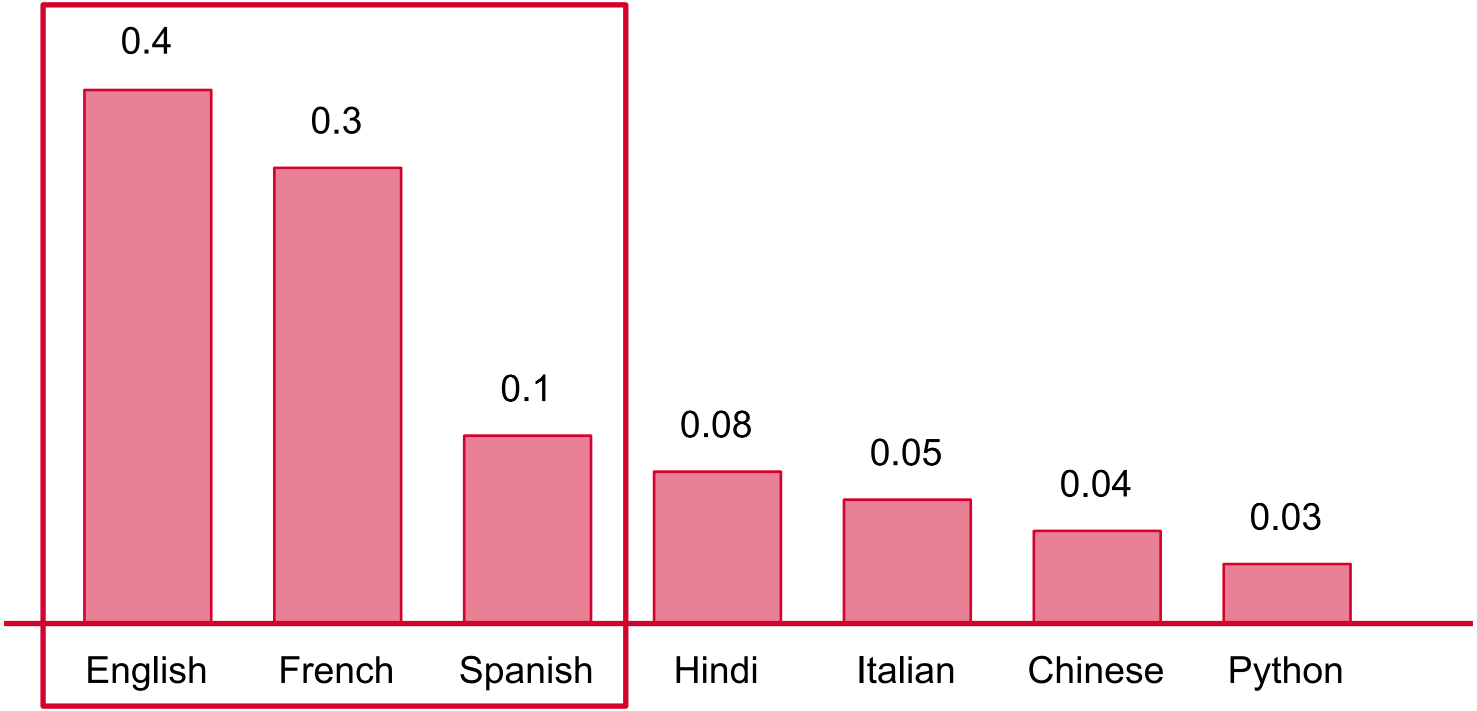


Top-p of 0.5

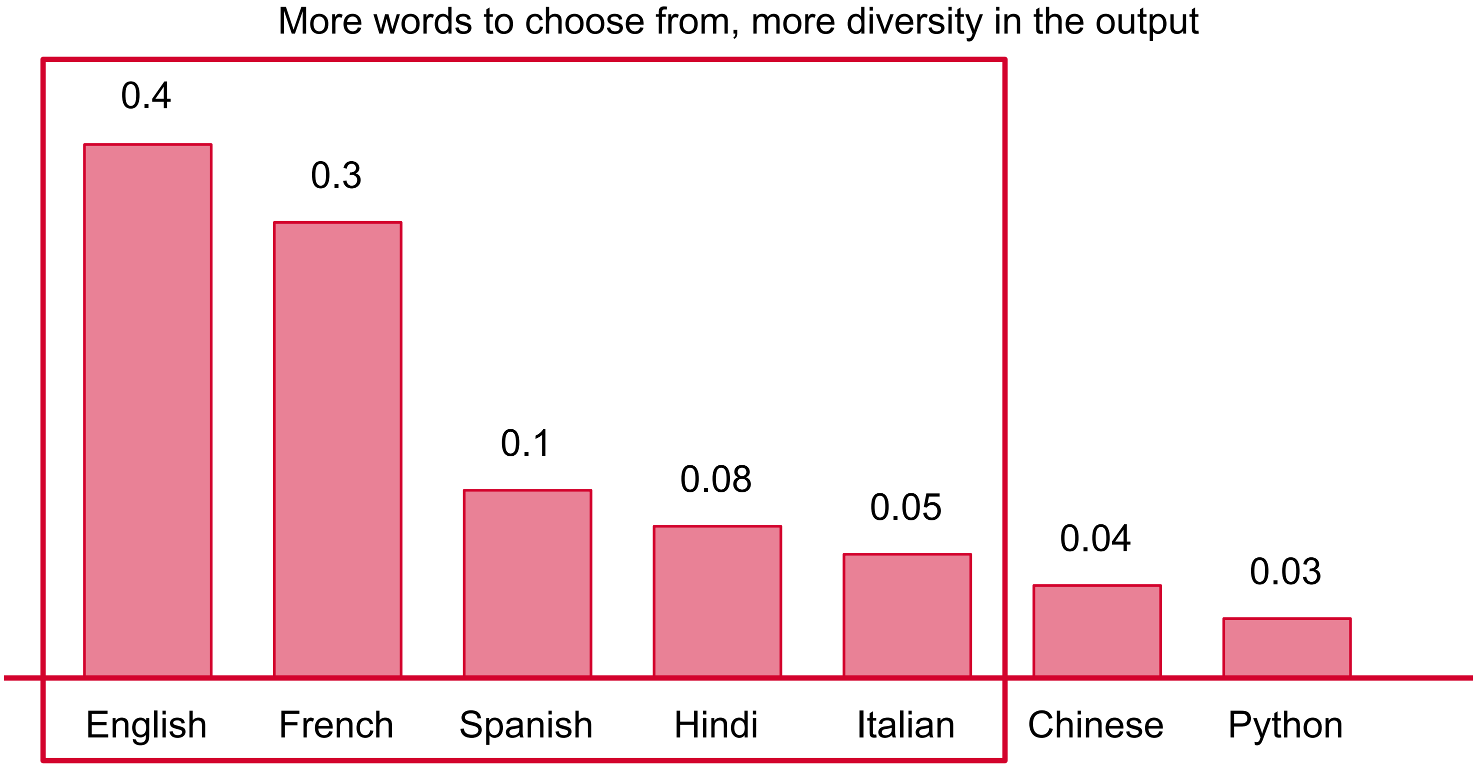


The next word selected will only choose between the smallest subset of words that exceeds the cumulative probability threshold

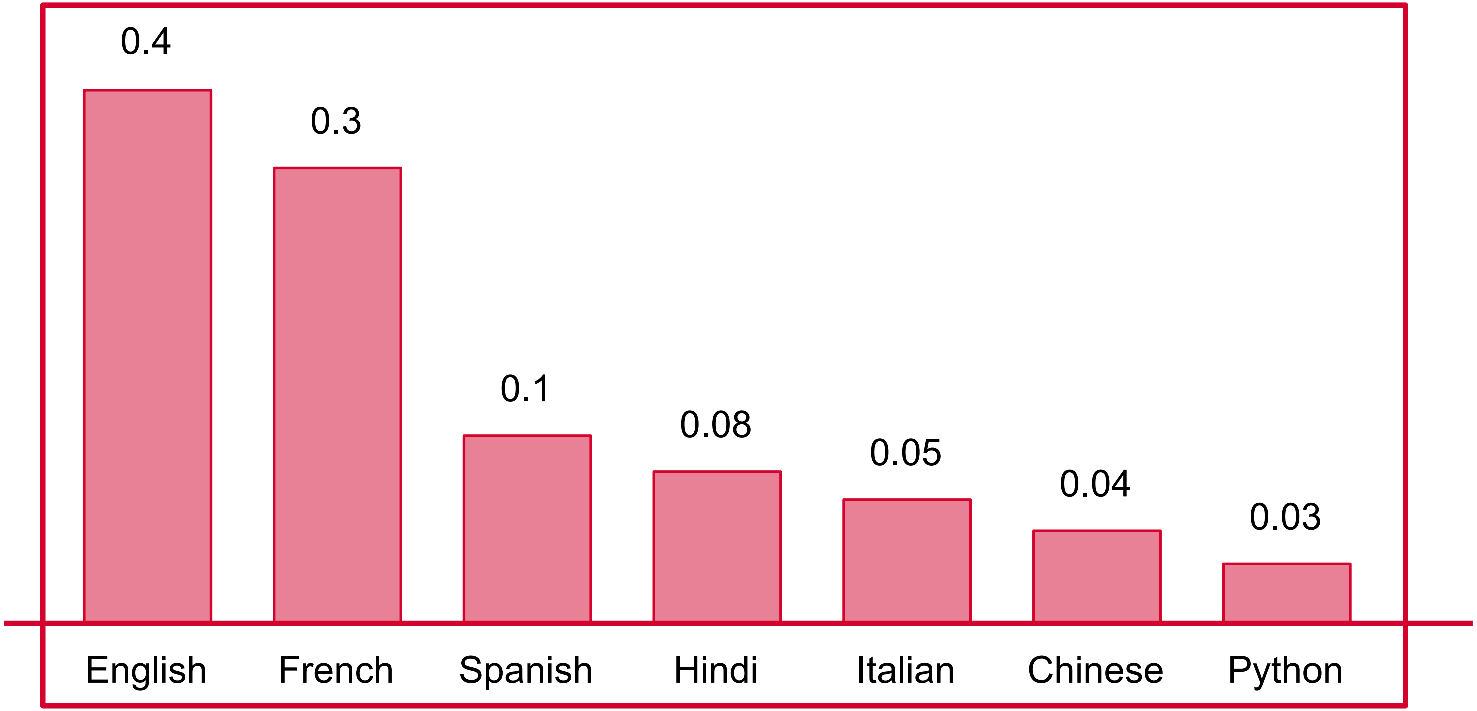
Top-p of 0.7


0.4
0.3
0.1
0.08
0.05
0.04
0.03
English
French
Spanish
Hindi
Italian
Chinese
Python

Top-p of 0.9


More words to choose from, more diversity in the output
0.4
0.3
0.1
0.08
0.05
0.04
0.03
English
French
Spanish
Hindi
Italian
Chinese
Python

Top-p of 1



Choose from among all possible words, greatest possible diversity