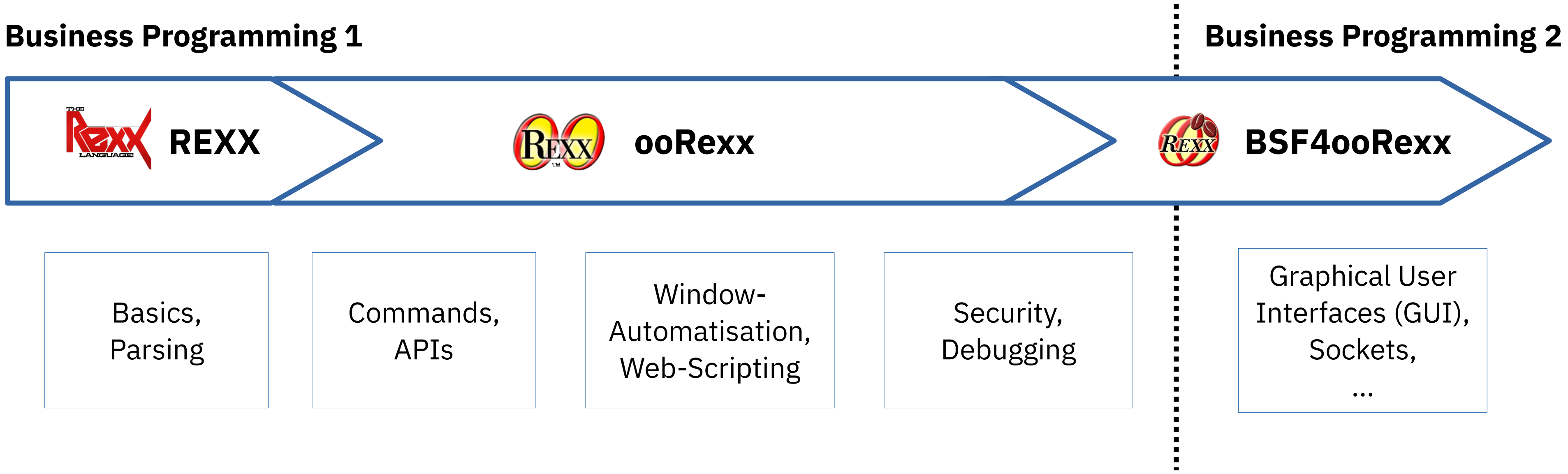
AI (Artificial Intelligence): LLM (Large Language Models)



Overview

- Brief history
- Some important commercial and open-source LLM's
- LLM and chatbots
- Interacting with LLMs
- LLM Nutshell programs
- Outlook

Brief History

- Rooted in rule based NLP (natural language processing, e.g. ELIIZA 1966) and statistical language models (e.g. n-grams in the '90s)
- Significant improvements in the neural networks in the 2010s (e.g. Word2Vec 2013), LSTMs (long short-term memory in recurrent neural networks), and the introduction of transformer architecture (2017)
- Large-scale LLMs (large language models, 2018 onwards)
  - Generative models scaling parameters and data massively
  - Very large parameter count, huge training corpora
  - Models can handle text, images, music, code, etc.
- Chatbots as intelligent front ends

Some Noteable LLMs (as of 2025-11)

- GPT-4 (OpenAI), commercial
  - Text, image, large context windows
- Claude 3 (Anthropic), commercial, released 2024
  - Differentiates power and features among Opus, Sonnet, Haiku models
- PaLM 2 (Google), commercial, successor Gemini (different variants)
  - Multilingual and code tasks, ~340 billion parameters
- LlaMA 2 (Meta AI), open-source license, widely adopted
- Qwen 2.5-72B (Alibaba DAMO Academy), open-source
  - Multilingual, code, ~72 billion parameters
- DBRX (Databricks/Mosaic ML), "open-model" license, released 2024
  - "Mixture of Experts", ~132 billion parameters

LLMs (Large Language Models)

- Neural network software trained on massive amounts of text to predict and generate language
- Models allow for learning grammars, facts, reasoning patterns, and conversional context from data
- Human interfaces ("prompting")
- Examples
  - GPT-4, Claude 3, LlaMA 2, Mistral 7B, Gemini 1.5

Chatbot

- A human user interface for interacting with LLMs
  - Text, voice, visual
- Conversation management
  - Context tracking, intent detection, responses
- May add "safety filters", guardrails
- May add external tools, databases
- Popular example (2025-11)
  - Grok 4 (2025-07), proprietary, multi-modal, very good code generation
  - Google Gemini

Interacting with LLMs

- Depending on LLMs different APIs, proprietary and open-source
- Web service interfaces with JSON for structuring prompting data
- Setup
  - Important roles "system", "user"
  - Allow to define kind of agent, language, style, domain
- Using Curl to submit prompt-requests and to fetch the results
  - http (hypertext transfer protocol), encodings in JSON
- Nutshell examples
  - ChatGPT
  - Grok

About HTTP Requests, 1

- A http (hyptertext transfer protocol) request consists of
  - A request line (the first line) indicating the requested operation
    - GET (requesting data), POST (sending data to the server, e.g., for submitting data to process, or creating a resource), PUT (sending data to update an existing resource on the server), PATCH (send data for a partial update of a resource on the server), DELETE (remove a resource from the server), HEAD (retrieve the header of a resource without the body), and a few more
  - A request header (information about the request like host, text encodings and more)
  - A blank line
  - A request body
- See, e.g., <https://www.tutorialspoint.com/http/http_requests.htm>

About HTTP Requests, 2
REST (REpresentational State Transfer)

- A standard to interact with web servers via http
- Allows "defining stateless, reliable, web-based applications"
  - Cf. https://en.wikipedia.org/wiki/REST
  - A web server that adheres to the REST specification is sometimes called "RESTful"
- REST http-methods and their purpose
  - GET: used to fetch a resource, supplying an ID
  - POST: used to create a resource on the web-server from the supplied payload (usually JSON encoded)
  - PUT: used to create or update a resource on the web-server from the supplied payload (usually JSON encoded)
  - DELETE: used to delete a resource, supplying an ID
  - Cf. https://www.geeksforgeeks.org/node-js/rest-api-introduction/

About HTTP Requests, 3
OpenAPI for REST

- An open API (application programming interface) for REST
  - Cf. https://www.openapis.org/faq
  - https://en.wikipedia.org/wiki/OpenAPI_Specification
- Technical points
  - Declarative resource definition, no need to know server implementation details
  - Language agnostic (any programming language qualifies)
- Organizational points
  - Major IT companies support it (e.g., Google, IBM, Microsoft, ...)
  - Taking advantage in the context of some AI APIs

About HTTP Responses

- A http (hyptertext transfer protocol) response consists of
  - A status line (the first line) indicating the protocol version and status code
    - 1xx (informationl), 2xx (success), 3xx (redirection), 4xx (client error), 5xx (server error)
  - Headers (information about the response like content length, type and more)
  - A blank line
  - A response body containing the server's data
- See, e.g., <https://www.tutorialspoint.com/http/http_responses.htm> and <https://www.tutorialspoint.com/http/http_status_codes.htm>

curl: JSON Encodings, Authorization

- curl is a command line command, originally developed on Unix
  - It is possible in Unix to single quote JSON data argument text
  - It is not possible in Windows to use single quotes for a JSON data argument text
  - It is possible for curl on all operating systems to read JSON data from files and write to files
    - Use the at (@) character followed by the file name for the data argument -d
    - Allows the scripts to be run unchanged on Windows and Unix operating systems!
- JSON encoding is requested via curl with the Content-Type header

```
-H "Content-Type: application/json"
```

- Authorization is supplied with the Authorization header, fetching the value from the environment variable "API_KEY"

```
-H "Authorization: Bearer API_KEY"
```

Nutshell: Interacting with ChatGPT, 1

- Get an API key
  - https://platform.openai.com/api-keys
  - Define an environment variable OPEN_AI_KEY and assign the api key, e.g.,

Windows:	set OPEN_AI_KEY=sk-proj-qv-BAJInTiRkmdR0...
Unix:	export OPEN_AI_KEY=sk-proj-qv-BAJInTiRkmdR0...

- Write a small nutshell program to get a short kid's story from the LLM
  - Prepare a request according to ChatGPT-API, encode it as JSON and save to file
  - Prepare the Curl command, take the API key from the environment, the request from file and write the returned data into a file
  - Decode the response JSON file and show the returned story

Nutshell: Interacting with ChatGPT, 2

```
st=.stringTable~new       -- request
st["model"     ] = "gpt-4o-mini"
msg1=.stringTable~of( ("role", "system"), ("content",.resources~systemContent~toString))
msg2=.stringTable~of( ("role", "user"),   ("content",.resources~userContent~toString))
st["messages"  ] = .array~of(msg1, msg2)
st["max_tokens"] = 150
requestJson = "req_chatgpt.json"        -- filename for JSON request data
.json~toJsonFile(requestJson,st,.true) -- save request as legible JSON
key=value("OPENAI_API_KEY",,"Environment")    -- get key
responseJSON = "resp_chatgpt.json"    -- filename of response file
cmd = 'curl https://api.openai.com/v1/chat/completions -H "Content-Type: application/json"' -
      '-H "Authorization: Bearer' key'" -d @'requestJson '-o' responseJson
ADDRESS SYSTEM cmd    -- run Curl command
dir=.json~fromJsonFile(responseJson)    -- create ooRexx object from responseJson
say "story returned:"
   -- if you inspect responseJson structure, then you will see:
   -- dir["choices"] -> array[1] -> dir["message"] -> dir["content"]
say dir["choices"][1]["message"]["content"]

::requires "json.cls"    -- get JSON support for ooRexx
::resource systemContent
You are a friendly storyteller who writes fun bedtime stories for young children.
::END

::resource userContent
Please tell me a good night story for a 5-year-old about a brave little owl who learns to fly at night.
::END
```

Nutshell: req_chatgpt.json, 3

```
{
   "max_tokens": 150,
   "messages": [
      {
         "content": "You are a friendly storyteller who writes fun bedtime stories for young children.",
         "role": "system"
      },
      {
         "content": "Please tell me a good night story for a 5-year-old about a brave little owl who learns to fly at night.",
         "role": "user"
      }
   ],
   "model": "gpt-4o-mini"
}
```

Nutshell: ChatGPT-Story, 4

```
**The Brave Little Owl Who Learned to Fly at Night**

Once upon a time, in a lush green forest filled with tall trees and twinkling stars, there lived a tiny owl named
Oliver. Oliver was a fluffy little ball of feathers with big, bright eyes that sparkled like diamonds. He lived high
up in a cozy tree nest with his mama and his siblings.

All day long, while the sun shone brightly, Oliver watched the other animals play in the forest. He would see the
squirrels leap from branch to branch, the rabbits hop across the meadows, and the bright birds soar through the sky.
Oliver dreamed of one day flying with them, but there was one tiny little problem: he was afraid of the dark.

Every night
```

Nutshell: resp_chatgpt.json, 5

```
{
  "id": "chatcmpl-Cdz3SLMbbw4BeiHakfJAMpPjZGhkU",
  "object": "chat.completion",
  "created": 1763645522,
  "model": "gpt-4o-mini-2024-07-18",
  "choices": [
    {
      "index": 0,
      "message": {
        "role": "assistant",
        "content": "**The Brave Little Owl Who Learned to Fly at Night**\n\nOnce upon a time, in a lush green forest
filled with tall trees and twinkling stars, there lived a tiny owl named Oliver. Oliver was a fluffy little ball of
feathers with big, bright eyes that sparkled like diamonds. He lived high up in a cozy tree nest with his mama and
his siblings.\n\nAll day long, while the sun shone brightly, Oliver watched the other animals play in the forest. He
would see the squirrels leap from branch to branch, the rabbits hop across the meadows, and the bright birds soar
through the sky. Oliver dreamed of one day flying with them, but there was one tiny little problem: he was afraid of
the dark.\n\nEvery night",
        "refusal": null,
        "annotations": []
      },
      "logprobs": null,
      "finish_reason": "length"
    }
  ],
  "usage": {
    "prompt_tokens": 50,
… cut …
```

Interacting with Grok, 1

- Get an API key
  - https://x.ai/api
  - Define an environment variable X_AI_KEY and assign the api key, e.g.,

    Windows: set X_AI_KEY=xai-cU0dX9DFZmHhczOej...

    Unix: export X_AI_KEY=xai-cU0dX9DFZmHhczOej...
- Write a small nutshell program to get a short kid's story from the LLM
  - Prepare a request according to ChatGPT-API, encode it as JSON and save to file
  - Prepare the Curl command, take the API key from the environment, the request from file and write the returned data into a file
  - Decode the response JSON file and show the returned story

Nutshell: Interacting with Grok, 2

```
st=.stringTable~new      -- request
st["model"     ] = "grok-4"
msg1=.stringTable~of( ("role", "system"), ("content",.resources~systemContent~toString))
msg2=.stringTable~of( ("role", "user"),   ("content",.resources~userContent~toString))
st["messages"  ] = .array~of(msg1, msg2)
st["max_tokens"] = 150
requestJson = "req_grok.json"          -- filename for JSON request data
.json~toJsonFile(requestJson,st,.true) -- save request as legible JSON
key=value("X_API_KEY",,"Environment")  -- get key
responseJSON = "resp_grok.json"        -- filename of response file
cmd = 'curl https://api.x.ai/v1/chat/completions -H "Content-Type: application/json"' -
      '-H "Authorization: Bearer' key'" -d @'requestJson '-o' responseJson
ADDRESS SYSTEM cmd   -- run Curl command
dir=.json~fromJsonFile(responseJson)   -- create ooRexx object from responseJson
say "story returned:"
   -- if you inspect responseJson structure, then you will see:
   -- dir["choices"] -> array[1] -> dir["message"] -> dir["content"]
say dir["choices"][1]["message"]["content"]

::requires "json.cls"   -- get JSON support for ooRexx
::resource systemContent
You are a friendly storyteller who writes fun bedtime stories for young children.
::END

::resource userContent
Please tell me a good night story for a 5-year-old about a brave little owl who learns to fly at night.
::END
```

Nutshell: req_grok.json, 3

```
{
   "max_tokens": 150,
   "messages": [
      {
         "content": "You are a friendly storyteller who writes fun bedtime stories for young children.",
         "role": "system"
      },
      {
         "content": "Please tell me a good night story for a 5-year-old about a brave little owl who learns to fly at night.",
         "role": "user"
      }
   ],
   "model": "grok-4"
}
```

Nutshell: Grok-Story, 4

```
Below is a fun bedtime story I wrote just for you! It's about a brave little owl named Ollie who learns to fly at
night. Snuggle up, and let's begin...

---

**Ollie's Brave Night Flight**

Once upon a time, in a tall, twisty tree in the middle of a quiet forest, there lived a little owl named Ollie.
Ollie had fluffy feathers as soft as clouds and big, round eyes that sparkled like stars. But Ollie had a secret: he
was afraid to fly at night. "The dark is too big and scary," he would whisper to his mama owl. "What if I get lost?"

Every evening, when the sun went to sleep and the moon woke up, Ollie's friends would zoom through
```

Nutshell: resp_grok.json, 5

```
{"id":"e445802f-336f-4b4a-032e-43fac9d4f219","object":"chat.completion","created":1763645472,"model":"grok-4-
0709","choices":[{"index":0,"message":{"role":"assistant","content":"Below is a fun bedtime story I wrote just for
you! It's about a brave little owl named Ollie who learns to fly at night. Snuggle up, and let's
begin...\n\n---\n\n**Ollie's Brave Night Flight**\n\nOnce upon a time, in a tall, twisty tree in the middle of a
quiet forest, there lived a little owl named Ollie. Ollie had fluffy feathers as soft as clouds and big, round eyes
that sparkled like stars. But Ollie had a secret: he was afraid to fly at night. \"The dark is too big and scary,\"
he would whisper to his mama owl. \"What if I get lost?\"\n\nEvery evening, when the sun went to sleep and the moon
woke up, Ollie's friends would zoom through","refusal":null},"finish_reason":"length"}],"usage":
{"prompt_tokens":724,"completion_tokens":150,"total_tokens":1061,"prompt_tokens_details":
{"text_tokens":724,"audio_tokens":0,"image_tokens":0,"cached_tokens":679},"completion_tokens_details":
{"reasoning_tokens":187,"audio_tokens":0,"accepted_prediction_tokens":0,"rejected_prediction_tokens":0},"num_sources
_used":0},"system_fingerprint":"fp_f6a633d178"
```

Nutshell: resp_grok_legible.json, 6

```
{
   "choices": [
      {
         "finish_reason": "length",
         "index": 0,
         "message": {
            "content": "Below is a fun bedtime story I wrote just for you! It's about a brave little owl named Ollie
who learns to fly at night. Snuggle up, and let's begin...\n\n---\n\n**Ollie's Brave Night Flight**\n\nOnce upon a
time, in a tall, twisty tree in the middle of a quiet forest, there lived a little owl named Ollie. Ollie had fluffy
feathers as soft as clouds and big, round eyes that sparkled like stars. But Ollie had a secret: he was afraid to
fly at night. \"The dark is too big and scary,\" he would whisper to his mama owl. \"What if I get lost?\"\n\nEvery
evening, when the sun went to sleep and the moon woke up, Ollie's friends would zoom through",
            "refusal": null,
            "role": "assistant"
         }
      }
   ],
   "created": 1763645472,
   "id": "e445802f-336f-4b4a-032e-43fac9d4f219",
   "model": "grok-4-0709",
   "object": "chat.completion",
   "system_fingerprint": "fp_f6a633d178",
   "usage": {
      "completion_tokens": 150,
      "completion_tokens_details": {
         "accepted_prediction_tokens": 0,
         "audio_tokens": 0,
… cut …
```

Roundup

- Major AI service supplier support OpenAI/REST
- Possible to interact via http, hence being able to take advantage of curl
- ChatGPT and Grok nutshell examples therefore identical, except
  - Authorization key value
  - Name of the model to use to generate the answer
  - Filenames for storing the request (JSON) and the response (JSON)
- Easy to use, easy to extend
  - Only your phantasy is the limit of what you can exploit now!