Big Data LDN

My Notes

How Databricks does Analytics and a whole lot more?

Chatbot to query high level datasets

Jupyter notebook instance connected to data

Tip and tricks for how to get better results from LLM+MCP conversations

Before it use to be prompt llm then reply. Now reply goes through agent first

Context is king!

Use markdown files to describe what you want to achieve. Very long context and describe all steps

Focus on error messages! Be descriptive and recovery messages on what to do next in error messages

Self healing loops (do iterations on format and quality then execute)

Blueprints are the new code!

Have MCP server read sql and write ai readable descriptions

Agent controller (master agent) have access to MCP

Agents of Change – Will AI Agents Be the Next Hype Bubble, or Transform Companies Forever?

Governance (stack traceability even integration to GitHub to see where the problem occurs)

Maia
Context files to generate pipelines (instead of)
Structure

Get the data and build something that works

Set up training (what it does and what it doesn’t do)

Writing detail prompts

Machine Scale vs. Human Scale: The Looming Crisis in Data Engineering

Maia

A2A

Exposes data from central team

Collaborative working (can generate pipelines that look like your code, also can work with pipelines already generated)

Senior engineer working like tech lead

Commit documentation

Agentic AI Architecture - how to use LLMs at enterprise scale

Fablio : text based role playing fantasy game

Genai platform architecture

Hidden technical debt in machine learning systems

GENAI platform: RAG
App. Prompt templates. Prompt construction. Prompt Argumentation. scoring. Post processing app

Genai platform: RAG
Wrap prompt with RAG

Metrics are important

Agent personas:

Tuned and trained to have a certain role

Agent consist of:
- brains (LLM)
- memory (Context - history of prompts/answers)
- tools (for actions)

Tell the agent what it is!

Role
Goal
Behaviour
Purpose

Agent AI architecture patterns
Have agents be critics etc
Refinement: critic
Prompt chaining: output of one goes into the next so splitting prompt to multi agent important
Routing:
Parallelisation:
Orchestrator: supervisor agent, gather results together summarise and give back to user

Autonomous, modular, loosely coupled, scalable, distributed
- microservices, object orientated, service oriented architecture

Cost
- inter agent communicate
- deployment
- cost

Best practices
- clear contracts
- enable agent discovery
- apply versioning, evolution communication of new versions
- monitoring
- allow polyglot language / systems

Agents are the new microservices

Can MCP be more than just the latest way to codify our stubbornly opinionated data egos?

Keboola

From Dev to MVP in Less Than 30 Days: Real-World Lessons from Databricks Engineers

Have strong narrative : 1 page slide so people can promote you MVP

Log blockers and risks

Think legislation, third party, proprietary software have measurable metrics

Ship core functionality first. Polish later

Time, revenue and risk (translate tech to business wins)

Regular check ups in finance legal security

FastMCP: Model Context Pragmatism

Prefect built FastMCP

Aaazzam VP of product

Why MCP:

Humans are orchestrators - we are the integration layer

Solving cross coordination problems

Nowadays the workplace is the LLM interfaces

Trying to pass c

Great for one shot atomic pass but not good context. They can only orchestrator what they can access

How do i bring context to LLM interfaces Trying Inject business logic to UI

We are trying to offload our tasks to LLM

Orchestrator requires
Workflows
Resources
Tools
Who is asking (state)

Context makes capabilities dynamic! What you can do depends on what information you can provide

Why not REST?
Fixed capabilities: stateless

For LLM you need runtime discovery, bidirectional (communicate state) (stateful)
Eg logging in changes state of what's available

MCP is orchestration protocol

They allow stateful bidirectional interaction (rpc) of things (capability discovery)

Protocol
Resource
Tools

One central contract instead of many plugins

FastMCP cloud - Prefect

What MCP doesn't have

Protocol: is opinionated about state

Ecosystem: client can't use what servers could build. MCP clients don't support prompts, resources etc

Deployments: static tool servers because dynamic is hard

1. State is hard to do. Priority externalising session state. Protocol built for single server serving single session

2. MCP compatible Clients lagging behind.
Prioritise building for client (tool that exposures resources)

3. Observability is hard. Track session ID (product analytics). Qwarg for user info, send client telemetry in future

4 dynamic context needs to be default

What's it has figured out

1. Burn token on description of server. Anti pattern is dumping your API. You are increasing decision space, you must reduce it

2. Hyper curating: cut down the decisions. Describe workflows and when to use it. Agent story vs User story. Curate obsessively. Dynamic name-spacing whilst clients peruse namespace

MCP Pattern
Internal expertise engines
1. Encode your top performers workflows

2. Cross system orchestration across different systems internally (solve 48 browser tab problem)
3. Investigate assistants
Cross system observability tools (failure can be help to MCP stuff)

Prefect is open source and could offload infrastructure work

Knowledge Graphs as a Reasoning Engine: Deploying Agents To Uncover Deep Insights in Your Connected Data

Reasoning is the cognitive process of deriving knowledge from evidence

LLM limited to data they have been trained on

RAG offers solution and retrieves info

Agents helpful to match them together

LLM that can do reason on graphs

GDS Agent: A graph Algorithmic Reasoning agent

- description of algo, match it up, run on database, results get

---

Small Language Models Beginners course

Small language model : at or below 12 billion parameters

SLM: specialist
LLM: generalist

Pros SLM:

- Cost, power (paper - large language models: significant cost)

—Developing agents:

Read Open AI has a practical guide to building agents

Start with baseline - what does good look like

Choose best model that does that

Read Nvidia small langue models are the future of agent ai

Keep large for orchestrator, small for agents (keep large for reasoning)

— quantization
Chose quantization level to save money too. That choose level of detail the weights are saved as. Sometimes less money, faster and better performance (good for inference endpoints)

E.g Q4 quantization (8 times smaller, good for edge devices, local)

Tip: don’t go below q4

— distillation
Have teacher model train student model. Can save money

— Choose your deployment pattern

What is edge compute?

— Local deployment
Oollama needs to be manually configured in windows

In oollama you have to create custom model file to obtain model of quantize level you want

— usage

After you create model, you can select and copilot

Can use continue.dev too

— Managed cloud platform

- data stays in tenant
- use AWS sagemaker and jumpstart

- use huggingface cli to download model in uc volume, use python wrapper (using transformers), declare unity catalog and deploy (serverless endpoint)

— cloud api access

- Pay by token
- could deprecate old api endpoints, be careful!

- using huggingface interface API playground

As context increases, token increases, money increases

Start with performance, baseline then challenger,

Always look at the license

Use the instruct version of the model if available

Let's talk about Context - Why AI Agents are not that smart!

Context is everywhere and can provide agent with a data that allows personalised experiences

Smart guardrails - time base and behavioural triggers that restricts AI GOING OFF

For operational reliability
-
