# System Engineering Aspects of End-to-End tinyML

Jan Ernst,
Director of AI

Latent AI
November 2021

# Latent AI – Offerings

Software to optimize AI models for low power and latency, faster time to market.

Deployment workflows for various hardware.

State of the art reference models for common tasks.

Tooling throughout ML lifecycle: from design exploration to enterprise deployment management.

White paper: https://latentai.com/unlocking-the-power-of-edge-ai/

## Awards

https://bit.ly/2EVXsBF

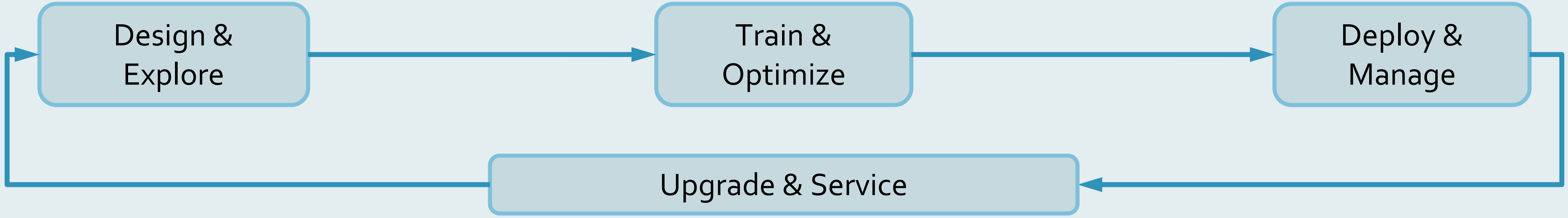
Top 100 AI Startup 2021

https://tcrn.ch/34juQee
https://vimeo.com/460028482

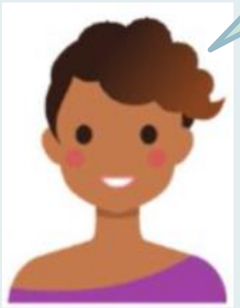
Top 60 Edge Computing Companies

## Investors

# ML Lifecycle



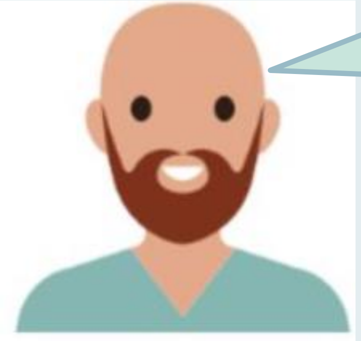

ML Expert Evelyn



I want to build cool stuff!

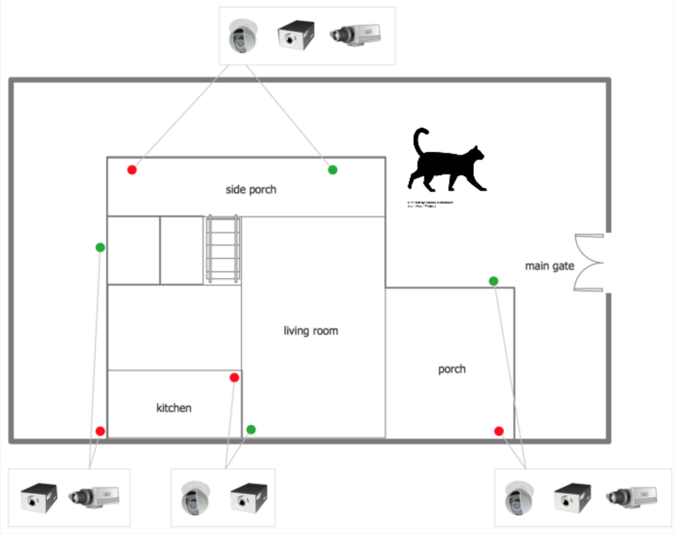
# The Journey of ML Expert Evelyn

Design & Explore → Train & Optimize → Deploy & Manage



Solutions Steve
(Links with customer)

Let's build a new application:
"Find my cat!"




Awesome! Tell me more about the requirements…

Expert Evelyn

LatentAI

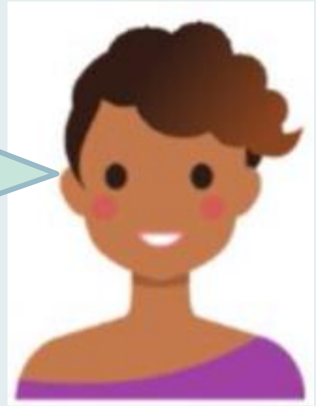
# The Journey of ML Expert Evelyn

**Design & Explore** → Train & Optimize → Deploy & Manage

What hardware fits into my power envelope?

What model fits into my hardware?

What data do I have/need?



# The Journey of ML Expert Evelyn

Design & Explore → **Train & Optimize** → Deploy & Manage

How do I train my model on my data?

How do I optimize my model for the hardware?

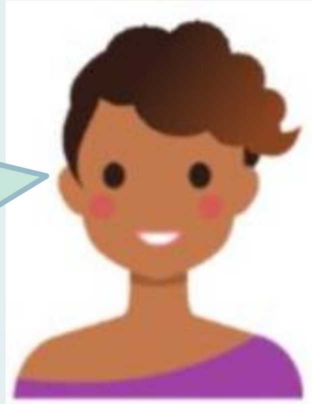
# The Journey of ML Expert Evelyn

Design & Explore → Train & Optimize → Deploy & Manage

How do I run my model on device?

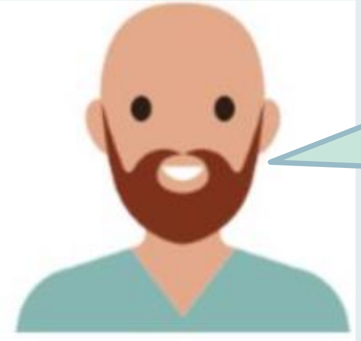
How do I integrate into a larger system?

What about enterprise reporting?



# The end of the Journey?

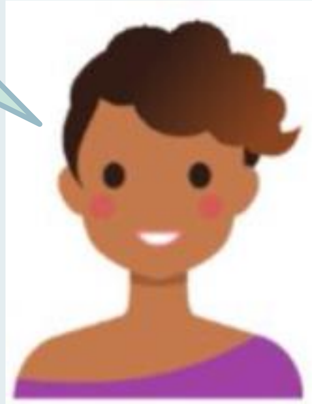
Design & Explore → Train & Optimize → Deploy & Manage



Solutions Steve

Awesome! I'll start integrating and selling this!

Here is the deployable component!



Expert Evelyn

# Somewhere down the road

Design & Explore → Train & Optimize → Deploy & Manage

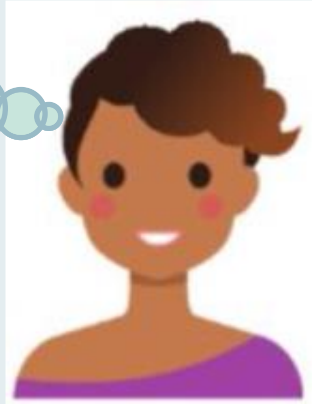
Solutions Steve

**“Find my cat!” is a hit!**

What about **dogs**?
Can we add **RFID**?
Need to recognize pet **identity**!
Can I use **cheaper** HW?
Let’s **voice-activate** our app!

Dogs? RFID?
Recognition?
HW? Voice?

**How do we scale this up?**



Expert Evelyn

# Retrospective

Design & Explore → Train & Optimize → Deploy & Manage

**Early opportunistic design choices** instead of **holistic optimization**

- "Choose model due to framework support"
- "Found nice git repo with model and training"

**Evelyn needs** best of breed, latest available state of the art from various ecosystems.

**Evelyn must** develop MxN combinations of model and optimization methods.

**Evelyn needs** to validate performance criteria of task vs. target HW.

**Evelyn must** replicate her accuracy evaluation code with the vendor API and guarantee that it's the same performance as in training phase.

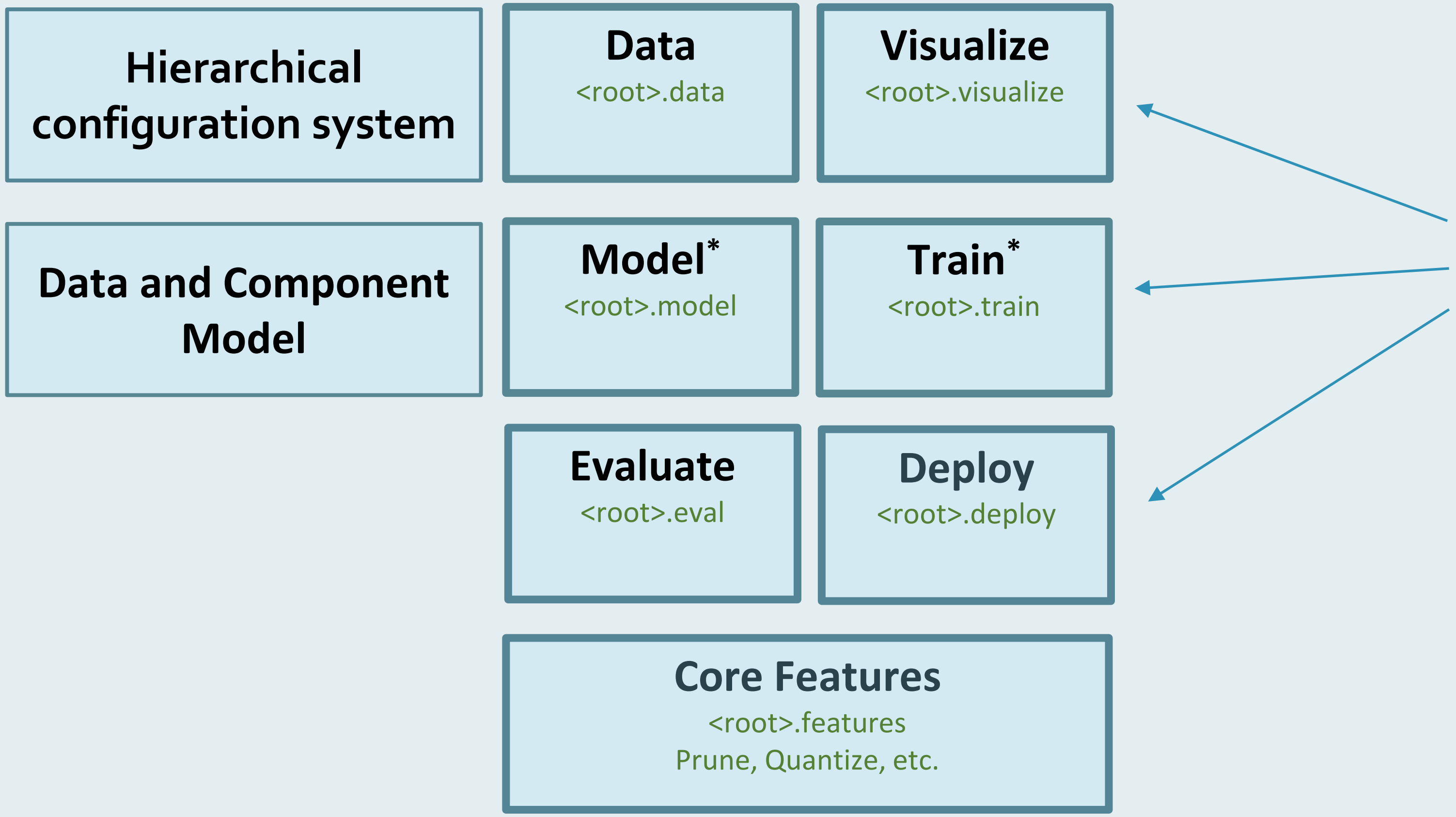
# How to improve?

**Apply Software Engineering design principles to ML Engineering**

- Compositionality and Interoperability
- Separation of concerns
- Autonomous components

**“System”**: thin, flexible shell of loose coupling of components

# Our approach – Compositional and Modular




**A Component is**

- Self-contained, independent
- Describes itself completely via interfaces and implied semantics
- Exposes its possible interactions with other components

*Pytorch

LatentAI

# System Perspective

**Data**
<root>.data

**Model**
<root>.model

**Train**
<root>.train

**Visualize**
<root>.visualize

**Evaluate**
<root>.eval

**Deploy**
<root>.deploy

**Core Features**
<root>.features
Quantize, Prune, Adapt, etc.

**Hierarchical configuration system**

**Data and Component Model**

**Expose two types of Interfaces:**

**"What am I?"**
- Images, Signals, Tensors
- Metadata, Annotations

**"How do I interact with other components?"**
- Augmentation
- Transparent caching

**"What does data represent?"**
- "Annotation is something a sample may have"
- "Annotations can have confidence, geometry, timestamp, …"
- "Bounding box is quadrilateral spatial geometry"

**"How can data be interpreted?"**
- "Object detections can be image labels"
- "Bounding box can be weak supervision for segmentation"

# System Perspective

Data
<root>.data

Model
<root>.model

Train
<root>.train

Visualize
<root>.visualize

Evaluate
<root>.eval

Deploy
<root>.deploy

Features
<root>.features
Prune, Throttle, Adapt, etc.

Hierarchical configuration system

Data Model

Resulting system is modular and compositional

Any component can be developed on its own.

A system is fully described by the set of its components.

Abstractions not in the way of **going deep anywhere**.

**The power of compositionality**: No-code system design (only YAML/CLI configuration)

Examples

| | | | | |
|---|---|---|---|---|
| **Data**<br><root>.data =<br>**Imagenet** | **Visualize**<br><root>.visualize =<br>**default** | | = | Visualize annotations |
| **Data**<br><root>.data =<br>**Imagenet** | **Model**<br><root>.model =<br>**Resnet50** | **Visualize**<br><root>.visualize =<br>**default** | = | Visualize model prediction |
| **Data**<br><root>.data =<br>**Imagenet** | **Model**<br><root>.model =<br>**Resnet50** | **Train**<br><root>.train =<br>**default** | = | Train image classifier |
| **Data**<br><root>.data =<br>**MSCOCO** | **Model**<br><root>.model =<br>**Yolo v5 - tiny** | **Train**<br><root>.train =<br>**default** | = | Train object detector |

## **The power of compositionality**: No-code system design (only YAML/CLI configuration)

**Data**
<root>.data =
**MSCOCO/dog**
**PASCAL/cat**

**Model**
<root>.model =
**Yolo|SSD|EffDet**

**Train**
<root>.train =
**default**

= Combine data, train and sweep multiple object detector families

**Data**
<root>.data =
**MSCOCO**

**Model**
<root>.model =
**Yolo v5 - tiny**

**Train**
<root>.train =
**default**

**Core Feature**
<root>.features =
**[quantize]**

**Evaluate**
<root>.eval =
**default|target**

**Deploy**
<root>.deploy =
**Raspberry_PI4B**

= Train a quantized object detector for RPi, evaluate compiled artifact via mAP on host and target

# How does it help Evelyn to scale?

| | | |
|---|---|---|
| Data<br><root>.data =<br>MSCOCO | Model<br><root>.model =<br>Yolo v5 - tiny | Train<br><root>.train =<br>default |
| Core Feature<br><root>.features =<br>[quantize] | Evaluate<br><root>.eval =<br>default | Deploy<br><root>.deploy =<br>Raspberry_PI4B |

**Faster time to market**: Can quickly design and sweep new variations and evaluate end to end (data to device).

**Simplified ML CICD**: Individual components or whole system, don't need expert on end-to-end pipeline.

**Simplifies ML Dev**: Improve parts without needing to understand whole pipeline: Separate Data, Model, Deploy teams.

**Simplifies ML Ops**: Reproducible processes, end-to-end tracking of data and model provenance.

**Trusted results**: Guaranteed same evaluation on host and target HW (preprocessing, postprocessing, protocol)

**Best practices** built-in or interoperable: distributed, mixed precision, fault tolerance, experiment tracking, etc.

# Thank You

Contact us at
info@latentai.com

www.latentai.com

Please fill out a tinyML survey

https://bit.ly/3GJ5etE

Latent AI is conducting a short survey to better understand how engineers and developers make design choices for their tiny ML systems.  We will share survey results with those that participate.