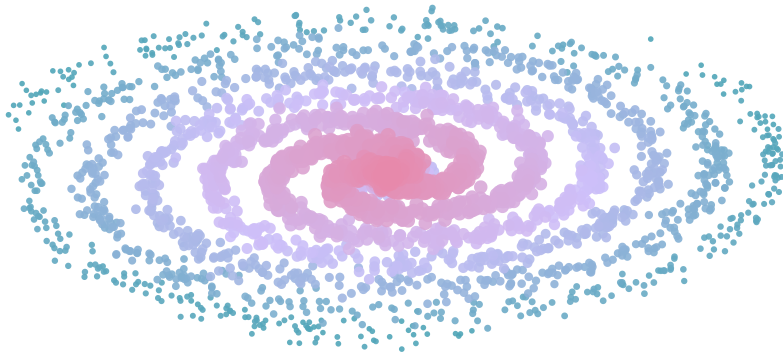


Bringing paper to life: A modern template for scientific writing



Publish-ready workflow that lets you focus on ideas, not infrastructure

AUTHOR

[Thibaud Frere](#)

AFFILIATION

[Hugging Face](#)

PUBLISHED

Sep. 01, 2025

Welcome to this open source research article template. It helps you publish clear, modern, and interactive technical writing with minimal setup.

Grounded in up to date good practices in web dev, it favors interactive explanations, clear notation, and inspectable examples over static snapshots.

Available on [GitHub](#) deployable on [Hugging Face Spaces](#).

FEATURES

Markdown-based

KaTeX math

Syntax highlighting

Academic citations

Footnotes

Table of contents

Mermaid diagrams

Plotly ready

D3.js ready

HTML embeds

Gradio app embeds

Dataviz color palettes

Optimized images

Lightweight bundle

SEO friendly

Automatic build

Automatic PDF export

Dark theme

Mobile friendly

Latex import

Template update system

Introduction

The web offers what static PDFs can't: interactive diagrams, progressive notation, and exploratory views that show how ideas behave. This template treats interactive artifacts—figures, math, code, and inspectable experiments—as first-class alongside prose, helping readers build intuition instead of skimming results.

Your article?

Build your research paper with this template

Getting Started

Installation

The recommended way is to duplicate this Space on Hugging Face rather than cloning it directly:

1. Open the Space: [🤖 science-blog-template](#) and click `Duplicate this Space`.
2. Give it a name, choose visibility, and keep the free CPU instance.
3. Clone your new Space repository.

```
1 | git clone git@hf.co:spaces/<your-username>/<your-space>
2 | cd <your-space>
```

4. Use Node.js 20 or newer.
To manage versions, consider using nvm

- macOS/Linux: see [nvm-sh](#)
- Windows: see [nvm-windows](#)

```
1 | nvm install 20
2 | nvm use 20
3 | node -v
```

5. Install lfs and pull files from the repository.

```
1 | git lfs install
2 | git lfs pull
```



If you attempt to push binary files without Git LFS installed, you will encounter an error.

6. Install dependencies.

```
1 | cd app
2 | npm install
```

And that's it!

You're ready to go! 🎉

Development

```
1 | npm run dev
```

Once started, the dev server is available at `http://localhost:4321`.

Build

```
1 | npm run build
```

Deploy

Every push automatically triggers a build and deploy on Spaces.

```
1 | # Make edits locally, then:
2 | git add .
3 | git commit -m "Update content"
4 | git push
```

Serving the `dist/` directory on any static host is enough to deliver the site.

A `[slugified-title].pdf` and `thumb.jpg` are also generated at build time.

You can find them in the public folder and point to them at `[domain]/public/thumb.jpg`.

Template Synchronization

Keep your project up-to-date with the latest template improvements. The sync system fetches the most recent changes from the official template repository at `https://huggingface.co/spaces/tfrere/research-article-template` and copies them to your project.

```
1 | # Preview what would be updated
2 | npm run sync:template -- --dry-run
3 |
4 | # Update template files (preserves your content)
5 | npm run sync:template
```

What gets preserved:

- Your content in `/src/content/`

What gets updated:

- All template files (components, styles, configuration)
- Dockerfile and deployment configuration
- Dependencies and build system

Writing your content

Once you have set up your project and started the development server, you can start writing your article.

Content structure

Your article lives in one and unique place. The `content` folder.



Article.mdx

The `article.mdx` file is the main entry point of your article which contains 2 main parts.



MDX is a mix of Markdown and HTML/JSX: write regular [Markdown](#) and [Components](#) when needed. We'll describe the available options you can use later in this guide.

FRONTMATTER

Metadata and options for the article. Each of them is described in the table below.

Frontmatter in `app/src/content/article.mdx`

```

1  ---
2  title: "This is the main title"
3  subtitle: "This will be displayed just below the banner"
4  description: "A modern, MDX-first research article template with math, citations, and
5  interactive figures."
6  published: "Feb 19, 2025"
7  tags:
8    - research
9    - template
10 authors:
11   - name: "Thibaud Frere"
12     url: "https://huggingface.co/tfrere"
13     affiliations: [1]
14   - name: "Alice Martin"
15     url: "https://example.com/~alice"
16     affiliations: [1, 2]
17   - name: "Robert Brown"
18     url: "https://example.com/~bob"
19     affiliations: [2]
20 affiliations:
21   - name: "Hugging Face"
22     url: "https://huggingface.co"
23   - name: "Example University"
24     url: "https://example.edu"
25 doi: 10.1234/abcd.efgh
26 licence: Diagrams and text are licensed under <a
27 href="https://creativecommons.org/licenses/by/4.0/" target="_blank" rel="noopener
28 norereferrer">CC-BY 4.0</a> with the source available on <a
29 href="https://huggingface.co/spaces/stfrere/research-article-template">Hugging Face</a>, unless
30 noted otherwise. Figures reused from other sources are excluded and marked in their captions
    ("Figure from ...").
31 seoThumbImage: "https://example.com/thumb.png"
32 tableOfContentsAutoCollapse: true
33 pdfProOnly: false
34 showPdf: true
35 ---

```

Frontmatter fields



CONTENT

Your story. Write your content here.

Content in `app/src/content/article.mdx`

```

1  import placeholder from '../assets/image/placeholder.png'
2  import Image from '../components/Image.astro'
3  import Sidenote from '../components/Sidenote.astro'
4
5  This paragraph is written in Markdown.
6
7  <Sidenote>
8    A short callout inserted via a component.
9  </Sidenote>
10 <Image src={placeholder} alt="Sample image with optimization" />
11
12 This paragraph is also written in Markdown.

```

CHAPTERS

If your article becomes too long for one file, you can organize it into separate chapters.

Simply create a new file in the `app/src/content/chapters` directory.
Then, include your new chapter in the main `article.mdx` like below.

Example

```
1 import MyChapter from './chapters/my-chapter.mdx';
2
3 <MyChapter />
```

You can see a living example here <app/src/content/chapters/best-practices.mdx>.

Table of contents

The Table of contents is generated automatically from your H2–H4 headings. Keep headings short and descriptive; links work on desktop and mobile.



You can make the table of contents collapse by changing the `tableOfContentsAutoCollapse` parameter in the `frontmatter`. Which is `true` by default.

Theme

All interactive elements (buttons, inputs, cards, etc.) are themed with the primary color you choose.

You can update this main color to match your brand by changing the `--primary-color` variable in the `app/src/styles/_variables.css` file.

Use the color picker below to see how the primary color affects the theme.

BRAND COLOR



Aerospace Orange

OKLCH 75.0%, 12.0%, 47°

RGB 236, 149, 106

#EC956A

HUE



47°

COLOR PALETTES

Here is a suggestion of color palettes for your data visualizations that align with your brand identity. These palettes are generated from your `--primary-color`.

Normal color vision — typical for most people



Categorical

For non-numeric categories; visually distinct colors.



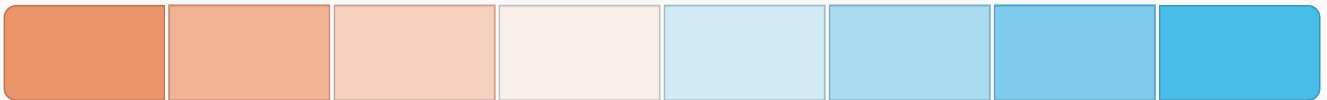
Sequential

For numeric scales; gradient from dark to light. Ideal for heatmaps.



Diverging

For numeric scales with negative and positive; Opposing extremes with smooth contrast around a neutral midpoint.



Use color with care. Color should rarely be the only channel of meaning. Always pair it with text, icons, shape or position. The simulation helps you spot palettes and states that become indistinguishable for people with color-vision deficiencies.

USING THE PALETTES

You can copy them manually from the palette viewer just above, or fetch colors via

`window.ColorPalettes.getColors(key, count)` where `key` is one of `'categorical'`, `'sequential'`, `'diverging'`, and `count` is the desired number of colors (defaults to 6).

Code example



Markdown

All the following markdown features are available natively in the `article.mdx` file. No imports needed, just write markdown directly:

Text formatting — `**Bold**` → **Bold**, `*italic*` → *italic*, `--strikethrough--` → ~~strikethrough~~

Code — ``inline code`` → `inline code`, triple backticks for code blocks

Lists — `- Item` for bullets, `1. Item` for numbered lists with nesting support

Links — `[text](url)` → [External links](#) and internal navigation

Highlight — `<mark>text</mark>` → **Highlighted text** for emphasis

See also the complete [Markdown documentation](#).

Advanced features — Explore specialized content types:

Math
LaTeX equations

Code
Syntax highlighting

Citation
Academic references

Footnote
Additional context

Mermaid
Diagrams & flowcharts

Separator
Visual breaks

Table
Data presentation

Audio
Sound embeds

Math

KaTeX provides full LaTeX math support with two simple syntaxes:

Inline math — Use `$...$` for equations within text: $x^2 + y^2 = z^2$

Block math — Use `$$...$$` for centered equations:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right) V$$

Advanced features — Aligned equations with IDs for cross-referencing:

$$\log p_\theta(\mathcal{D}) = \log \sum_{i=0}^N p_\theta((o, a)_i)$$

(1)

$$= \log \sum_{i=0}^N \int_{\text{supp}(Z)} p_\theta((o, a)_i | z) p(z)$$

(2)

$$= \log \sum_{i=0}^N \int_{\text{supp}(Z)} \frac{q_\theta(z | (o, a)_i)}{q_\theta(z | (o, a)_i)} \cdot p_\theta((o, a)_i | z) p(z)$$

(3)

$$= \log \sum_{i=0}^N \mathbb{E}_{z \sim p_\theta(\bullet | (o, a)_i)} \left[\frac{p(z)}{q_\theta(z | (o, a)_i)} \cdot p_\theta((o, a)_i | z) \right],$$

(4)

You can reference equations with links like [this equation](#).

Code example

Code

Use inline code with backticks ``...`` or ````` fenced code blocks ````` with a language for syntax highlighting (e.g., ``python``).

As an example, here is inline code: `greet("Astro")` and below is a block.

```
1 def greet(name: str) -> None:
2     print(f"Hello, {name}!")
```

Code example



Code output

If you want to display the output of a code block, you can use the `:::output` directive. If it's directly below the code block, it will adapt to the code block's styling.

```
1 def greet(name: str) -> None:
2     print(f"Hello, {name}!")
3
4 greet("Astro")
```

OUTPUT

Hello, Astro!

Or it can also be used at a standalone block.

OUTPUT

Hello i'm a standalone output block.

It also works in an accordion



Code example



Citation

The citation keys come from `app/src/content/bibliography.bib`.

Citation use the `@` syntax (e.g., `[@vaswani2017attention]` or `@vaswani2017attention` in narrative form) and are automatically collected to render the bibliography at the end of the article.

1. In-text citation with brackets: ([Vaswani et al., 2017](#)).
2. Narrative citation: As shown by [Kingma & Ba \(2015\)](#), stochastic optimization is widely used.
3. Multiple citations and a footnote together: see ([He et al., 2016](#); [McKinney, 2017](#)) for related work.
4. All citations in one group: ([Cover & Thomas, 2006](#); [Doe, 2020](#); [He et al., 2016](#); [Kingma & Ba, 2015](#); [McKinney, 2017](#); [OpenAI, 2023](#); [Pedregosa et al., 2024](#); [Raffel et al., 2020](#); [Silver et al., 2017](#); E. [Smith et al., 2024](#); J. [Smith et al., 2021](#); [Vaswani et al., 2017](#)).

Code example



You can change the citation style in the `astro.config.mjs` file. There are several styles available: `apa`, `vancouver`, `harvard1`, `chicago`, `mla`. Default is `apa`.

Footnote

Footnote use an identifier like `[^f1]` and a definition anywhere in the document, e.g., `[^f1]: Your explanation`. They are numbered and listed automatically at the end of the article.

1. Footnote attached to the sentence above [1](#).
2. Multi-paragraph footnote example [2](#).
2. Footnote containing a list [3](#).
3. Footnote with an inline code and an indented code block [4](#).
4. Footnote that includes citation inside [5](#) and another footnote [1](#).
5. Footnote with mathematical expressions [6](#).

Code example

Referencing

In research articles, you may have to make references to anything. They are basically html anchors. They can be used internally in the article or externally in other articles.

1. Title
Each title is automatically generated with a slugged version from the citation key. (slugged title from the citation key) like for example, the id `#mermaid-diagrams` is generated from the `Mermaid diagrams` title.

Example [Mermaid diagrams](#)

2. Image and chart

You can make a link to an image or a chart by adding an ID on it.

```
<HtmlEmbed id="neural-network-mnist-like"/>
```

 then you can link to it with a link like

```
<a href="#neural-network-mnist-like">Fig 1</a>
```

.

Example [Chart 1](#) or [Fig 1](#)

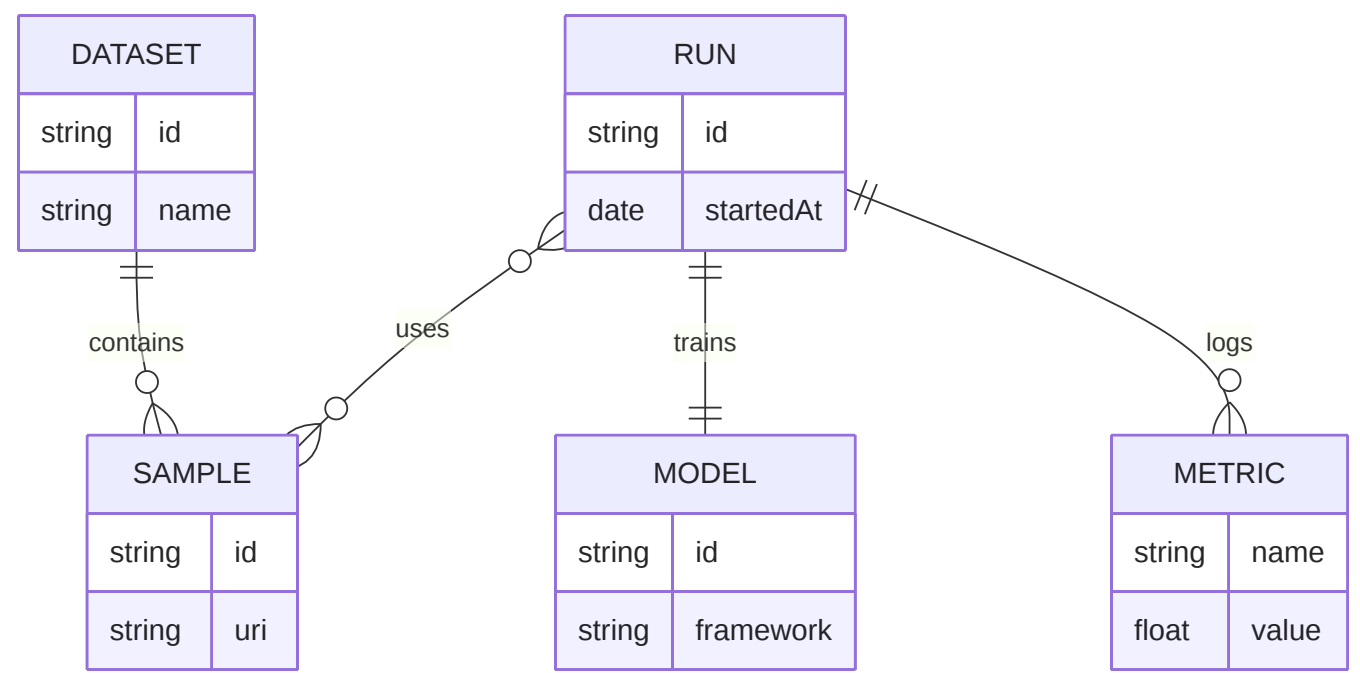


Available with: `Reference`, `Image`, and `HtmlEmbed` components all support the `id` prop for creating referenceable anchors.

Code example

Mermaid diagram

Native mermaid diagrams are supported (use a ````mermaid```` code fence). You can use the [live editor](#) to create your diagram and copy the code to your article.



Code example

Separator

Use `---` on its own line to insert a horizontal separator between sections. This is a standard Markdown “thematic break”. Don’t confuse it with the `---` used at the very top of the file to delimit the frontmatter.

Code example

Table

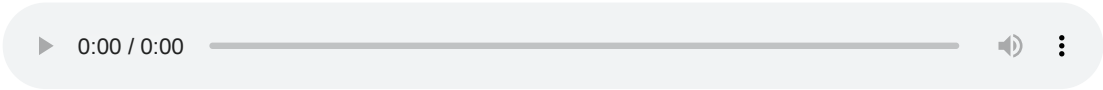
Use pipe tables like `| Column |` with header separator `| --- |`. You can control alignment with `: ---` (left), `: --- :` (center), and `--- :` (right).

Model	Accuracy	F1-Score	Training Time	Status
BERT-base	0.89	0.89	2.5h	✓
RoBERTa-large	0.92	0.92	4.2h	✓
DeBERTa-v3	0.94	0.94	5.8h	✓
GPT-3.5-turbo	0.91	0.91	0.1h	✓

Code example

Audio

Embed audio using `<audio controls src={...} />`.



Code example ▼

Components

All the following components are available in the `article.mdx` file. You can also create your own components by creating a new file in the `/components` folder.

How to import components

To use any component in your MDX file, add the import statement at the top:

```
1 import Image from '../components/Image.astro';
2 import Note from '../components/Note.astro';
3
4 # Your content
5
6 <Image src={myImage} alt="Description" />
7 <Note>This is a note</Note>
```

Here are the components that are available:

Image Optimized images	Placement Layout helpers	Accordion Collapsible content	Note Contextual callouts
Quote Elegant citations	HtmlEmbed External content	Reference Flexible content wrapper	Iframe Web embeds
Glossary Interactive term definitions	Stack Flexible grid layouts		

Image

Images automatically generate optimized `srcset` and `sizes` so the browser downloads the most appropriate file for the current viewport and DPR. You can also request multiple output formats (e.g., AVIF, WebP, fallback PNG/JPEG) and control lazy loading/decoding for better performance.

Figure 2 · A placeholder image description
Credit: [RCA Indian Head Test Pattern](#)

Prop	Required	Description
<code>zoomable</code>	No	Adds a zoomable lightbox (Medium-like).
<code>downloadable</code>	No	Adds a download button to fetch the image file.
<code>loading="lazy"</code>	No	Lazy loads the image.
<code>caption</code>	No	Adds a caption and credit.
<code>id</code>	No	Adds an <code>id</code> to the outer figure for deep-linking and cross-references.

Code example

▼

Placement

Use these helpers when you need to step outside the main content flow: Sidenotes for contextual side notes, Wide to extend beyond the main column, and Full-width for full-width, immersive sections.

SIDENOTES

This paragraph presents a key idea concisely.

This paragraph discusses the relationship between variables in a mathematical context, where we consider the function $f(x) = x^2 + 2x + 1$.

Code example

WIDE EXAMPLE

demo wide

Code example

FULL-WIDTH EXAMPLE

demo full-width

Code example

Reference

The Reference component provides a flexible wrapper for any content with an ID and HTML caption. It’s perfect for creating numbered references, cross-references, or any content that needs to be referenced throughout your article.



Note: The `Image` and `HtmlEmbed` components also support the same `id` and `caption` props for consistent referencing across your article.

you content here...

Figure 3 · Example reference with HTML caption and *styling*

Prop	Required	Type	Description
<code>id</code>	Yes	string	Unique identifier for the reference (used for deep-linking)
<code>caption</code>	Yes	string	HTML caption displayed below the content (supports HTML tags)

Code example



Use cases:

- Numbered figures with custom content
- Cross-references that can be linked to from anywhere in the article
- Flexible content blocks that need consistent styling and captioning
- Tables, charts, or any content that requires an ID and description

Accordion

Can be used like this `<Accordion>some content</Accordion>`. You can pass any children content.

What is an accordion?



The accordion component provides a collapsible content area that helps organize information efficiently. It's perfect for creating expandable sections, FAQ entries, or any content that benefits from progressive disclosure. Users can click the title to toggle the visibility of the content inside.

A table inside an accordion



Code example



Note

Small contextual callout for tips, caveats, or emphasis.



Heads-up

Use notes to surface context without breaking reading flow.

Operation completed successfully.

Be careful: this action cannot be undone.

Plain note without header. Useful for short clarifications.

Prop	Required	Type	Description
<code>title</code>	No	string	Short title displayed in header
<code>emoji</code>	No	string	Emoji displayed before the title
<code>class</code>	No	string	Extra classes for custom styling
<code>variant</code>	No	'neutral'	'info'

Code example

Quote

Elegant quotes with optional source attribution.

Backpropagation allows neural networks to discover their own internal representations of data.

Geoffrey Hinton, [Learning representations by back-propagating errors](#)

Prop	Required	Type	Description
<code>source</code>	No	HTML	Quote source/author (supports HTML links)

Code example

Glossary

The Glossary component creates interactive term definitions with hover tooltips. Perfect for technical terms, acronyms, and concepts that need explanation without breaking the reading flow.

Machine Learning Neural Network API

Prop	Required	Type	Description
<code>term</code>	Yes	<code>string</code>	The word or term to define
<code>definition</code>	Yes	<code>string</code>	The definition of the term
<code>class</code>	No	<code>string</code>	Optional CSS class to apply to the term
<code>style</code>	No	<code>string</code>	Optional inline style to apply to the term
<code>position</code>	No	<code>'top' 'bottom' 'left' 'right'</code>	Tooltip position (default: 'top')
<code>delay</code>	No	<code>number</code>	Delay before showing tooltip in ms (default: 300)
<code>disableOnMobile</code>	No	<code>boolean</code>	Disable tooltip on mobile devices (default: false)

Use cases:

- Technical terms that need explanation
- Acronyms and abbreviations
- Domain-specific concepts for broader audiences
- Interactive glossaries for educational content

Stack

The Stack component provides flexible grid layouts for organizing content. Perfect for comparisons, side-by-side examples, multi-column layouts, and responsive content organization.

AUTO LAYOUT

This layout automatically adjusts based on content and available space.

RESPONSIVE

Items will wrap to new lines as needed, with a minimum width of 300px.

FLEXIBLE

Perfect for varying content lengths and dynamic layouts.

Figure 4 · Image 1: Example of using Image component within Stack

Figure 5 · Image 2: Side-by-side comparison layout

Prop	Required	Type	Description
layout	No	"2-column" "3-column" "4-column" "auto"	Grid layout for the content (default: "2-column")
gap	No	"small" "medium" "large" string	Gap between items - predefined size or custom
class	No	string	Optional CSS class to apply to the wrapper
id	No	string	Optional ID for the stack

Code example

Iframes

You can embed external content in your article using iframes. For example, Gradio or even Github code embeds can be used this way.

Gradio embed example



Code example



HtmlEmbed

The main purpose of the `HtmlEmbed` component is to embed a Plotly or D3.js chart in your article. Libraries are already imported in the template.

They exist in the `app/src/content/embeds` folder.

For researchers who want to stay in Python while targeting D3, the [d3blocks](#) library lets you create interactive D3 charts with only a few lines of code. In 2025, D3 often provides more flexibility and a more web-native rendering than Plotly for custom visualizations.

This is a chart title

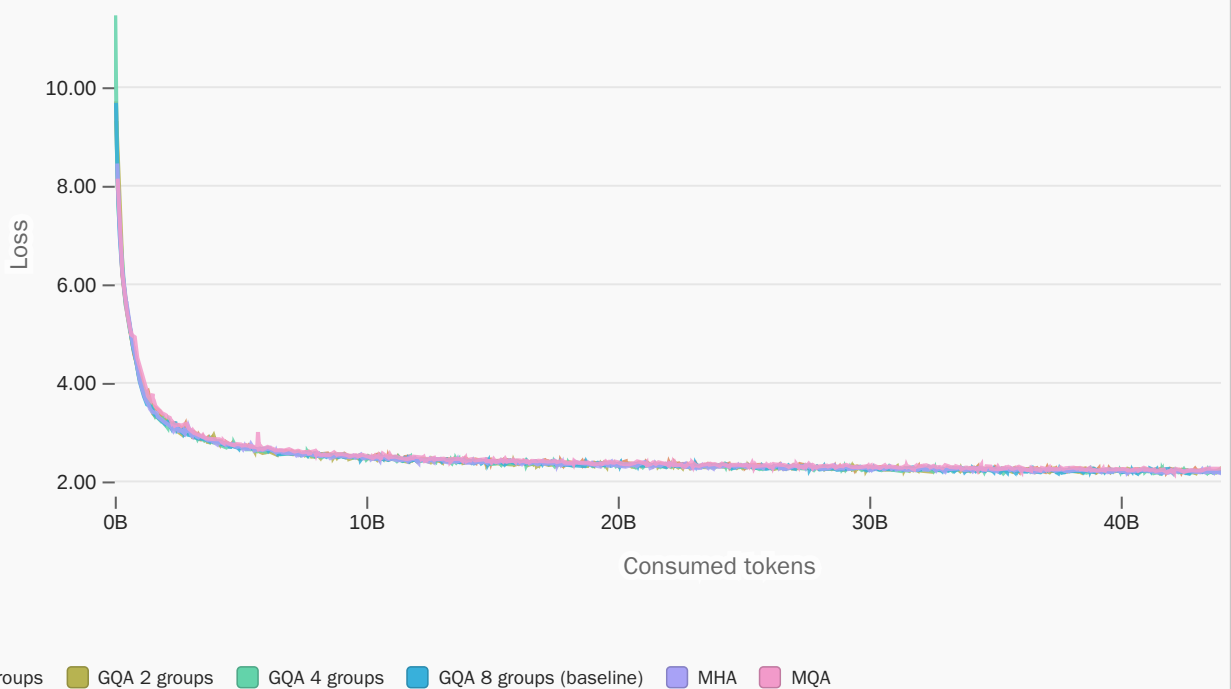


Figure 6 · Figure X: Some chart description

Credit: [Example](#)

Prop	Required	Description
<code>src</code>	Yes	Path to the embed file in the <code>embeds</code> folder.
<code>title</code>	No	Short title displayed above the card.
<code>desc</code>	No	Short description displayed below the card. Supports inline HTML (e.g., links).
<code>frameless</code>	No	Removes the card background and border for seamless embeds.
<code>align</code>	No	Aligns the title/description text. One of <code>left</code> (default), <code>center</code> , <code>right</code> .
<code>id</code>	No	Adds an <code>id</code> to the outer figure for deep-linking and cross-references.
<code>data</code>	No	Path (string) or array of paths (string[]) to data file(s) consumed by the embed.
<code>config</code>	No	Optional object for embed options (e.g., <code>{ defaultMetric: 'average_rank' }</code>).

Code example

DATA

If you need to link your HTML embeds to data files, there is an `assets/data` folder for this. As long as your files are there, they will be served from the `public/data` folder. You can fetch them with this address:
`[domain]/data/your-data.ext`

 Be careful, unlike images, data files are not optimized by Astro. You need to optimize them manually.

Dataviz page

The template includes a dedicated `/dataviz` page that automatically collects and displays all visual assets from your article in one place. It's useful for reviewing your figures, exporting them, or sharing a visual summary of your paper.

What it extracts:

Type	Source	Details
Charts	<code><HtmlEmbed></code> components	Rendered live with all props (config, data, wide)
Images	<code><Image></code> components	Resolved from MDX imports to actual files
Stacks	<code><Stack></code> containing <code><Image></code>	Grouped as a single item, preserving layout
Tables	Markdown tables (<code> ... </code>)	Parsed with headers and rows

The page is accessible at `/dataviz` and displays each item with:

- A type badge (Chart, Image, Stack, Table)
- A sequential number (#1, #2, ...)

- The title or caption when available
- A “View in article” link back to the source

Excluding items from the gallery:

Add the `skipGallery` prop to any `<Image>` or `<HtmlEmbed>` you don't want on the dataviz page. For `<Stack>` components, if all images inside have `skipGallery`, the entire stack is excluded.

Code example



Downloading charts:

Each chart on the dataviz page has a download button (PNG export). You can also batch-export all charts as a ZIP by running this in the browser console:

```
1 window.exportAllEmbeds()
```

Vibe coding charts



This is a work in progress. It may change quickly.

Prompt

This page explains how to use the directives to author D3 charts as self-contained HTML fragments. Using claude code works better.

The goal is to make responsive, accessible, interactive and dark mode ready charts.

1. The embed directives are available as a Cursor skill in `.cursor/skills/create-html-embed/` (the agent picks it up automatically). Full conventions: `.cursor/skills/create-html-embed/directives.md`.
2. Opt: use an already existing chart as a starting point.
3. Ask claude to code the chart. Here's a typical prompt:

```
1 I want you to code a new d3 chart named `yourchart`.
2 I have one CSV file called `yourdata.csv` in the data folder.
3 The csv has the following columns: `x`, `y`, `z`.
4 I want you to code a d3 chart that visualizes the data.
```

4. Once the chart created, iterate with little adjustments to make it better.

5. And that's it! 🎉

Base examples

These are fundamental chart types that serve as building blocks for more complex visualizations. You can find all the source code in `app/src/content/embeds`.

Post-Training Adventure

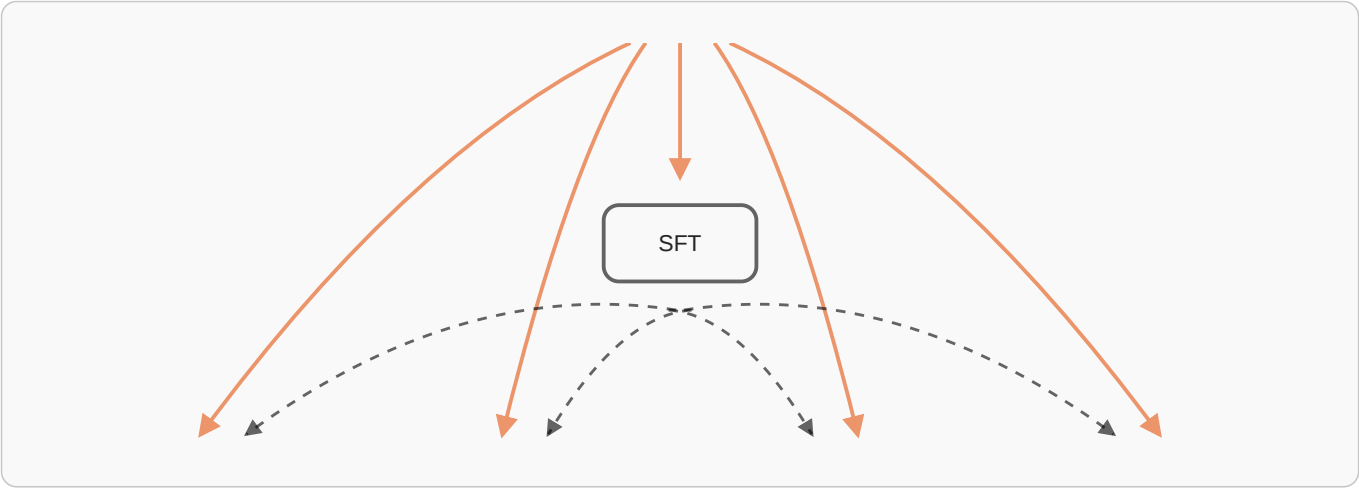


Figure 7 · Interactive visualization of a model's post-training journey, showing the different stages and decisions.

Advanced examples

These are more complex, interactive visualizations that demonstrate advanced D3 capabilities and real-world applications.

The Experiment Map — Source × Prompt × Model

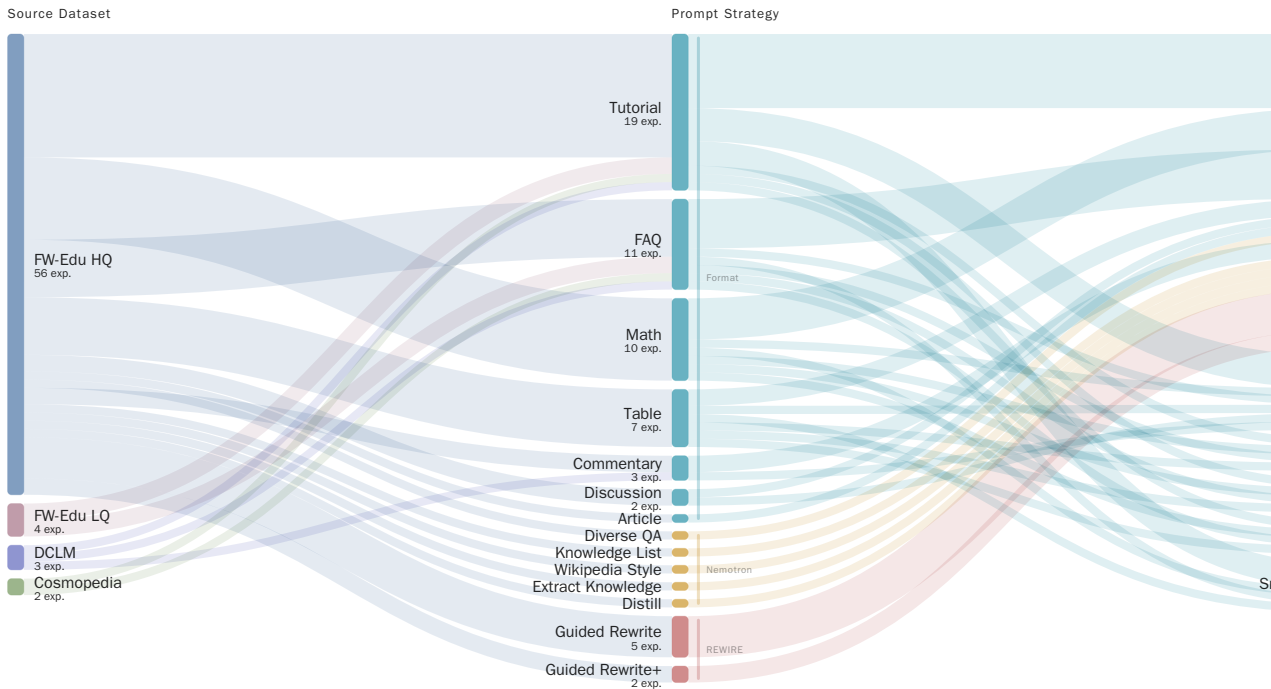
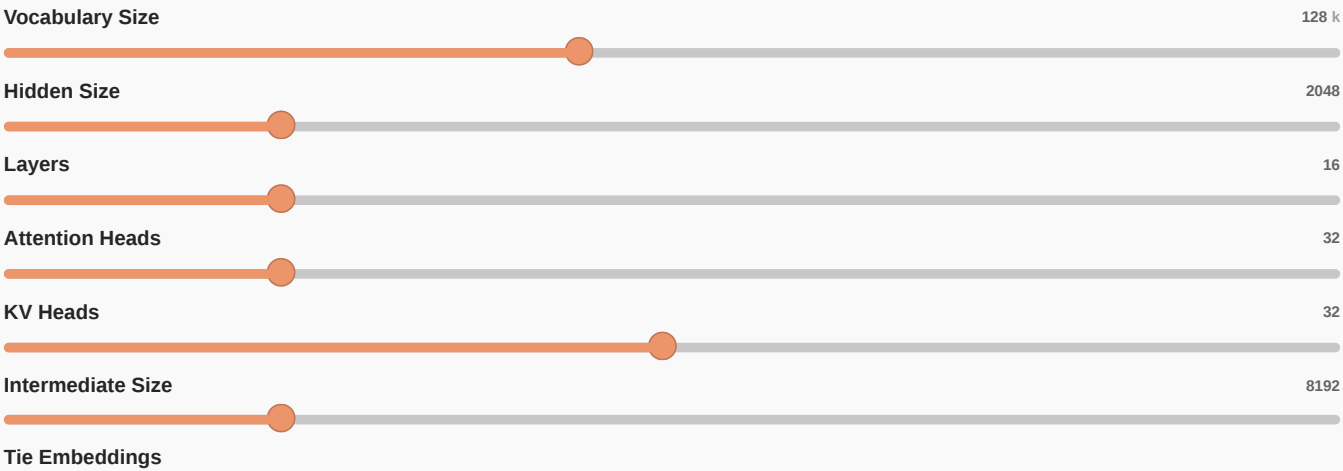


Figure 8 · Sankey diagram showing the flow of 65 experiments through source dataset, prompt strategy, and model family. Width = number of experiments.



Yes



1.34 B

Parameters

EMBEDDINGS

Input + Output Projection

262 M

128 k × 2048

$vocab_size \times hidden_size$

ATTENTION LAYERS

Q, K, V, O projections

268 M

16 × 2048² × 4

$layers \times hidden_size^2 \times 4$

FEED FORWARD

Up, Gate, Down projections

805 M

16 × 2048 × 8192 × 3

$layers \times hidden_size \times intermediate_size \times 3$

LAYER NORMS

Input + Attention norms

68 K

16 × 2048 × 2

layers × hidden_size × 2

ATTENTION TYPE

MHA

EMBEDDING STRATEGY

Tied

PARAMS PER LAYER

67 M

Figure 9 · Interactive parameter calculator to visualize the parameter distribution of a dense transformer model. Useful for making architectural decisions or configuring ablations.

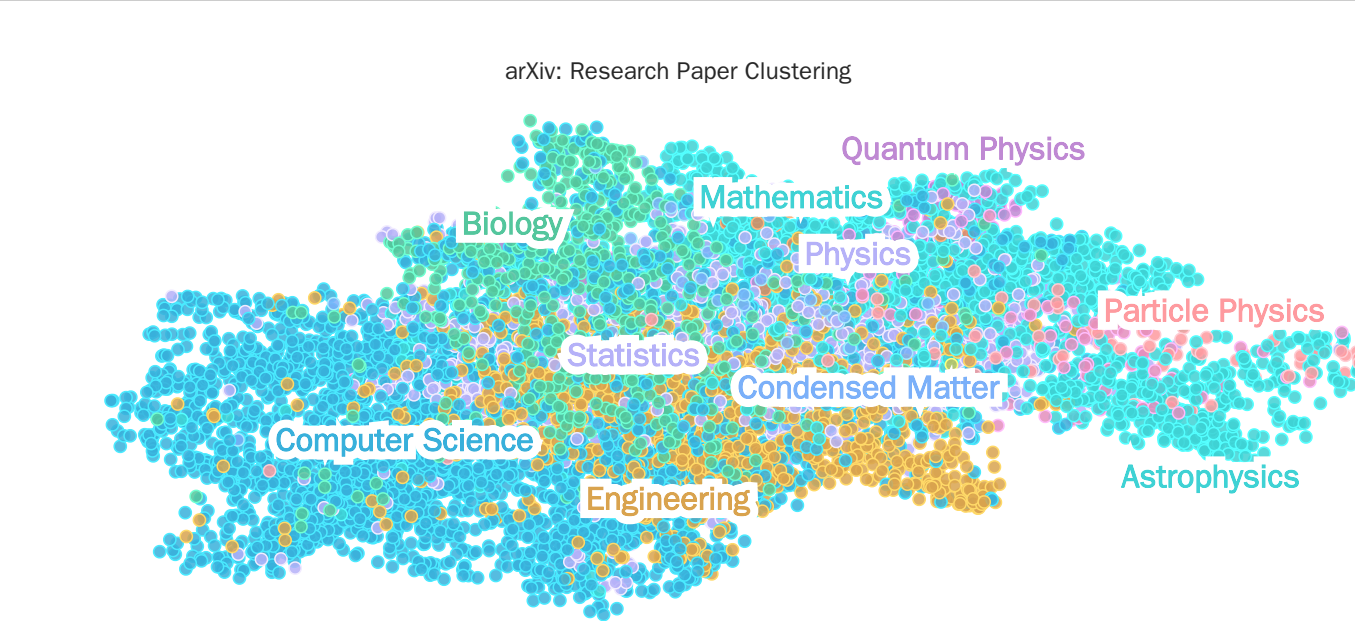


Figure 10 · Interactive visualization of ~8k recent arXiv submissions via UMAP dimensionality reduction. Each point represents a research paper positioned by semantic similarity. Colors indicate academic categories (cs.AI, cs.LG, cs.CV, etc.).

Visual Similarity of Typefaces



Figure 11 · Interactive 2D visualization of 382 Google Fonts clustered by visual similarity via UMAP. Each point represents a typeface positioned based on pixel-level differences computed from font matrices.

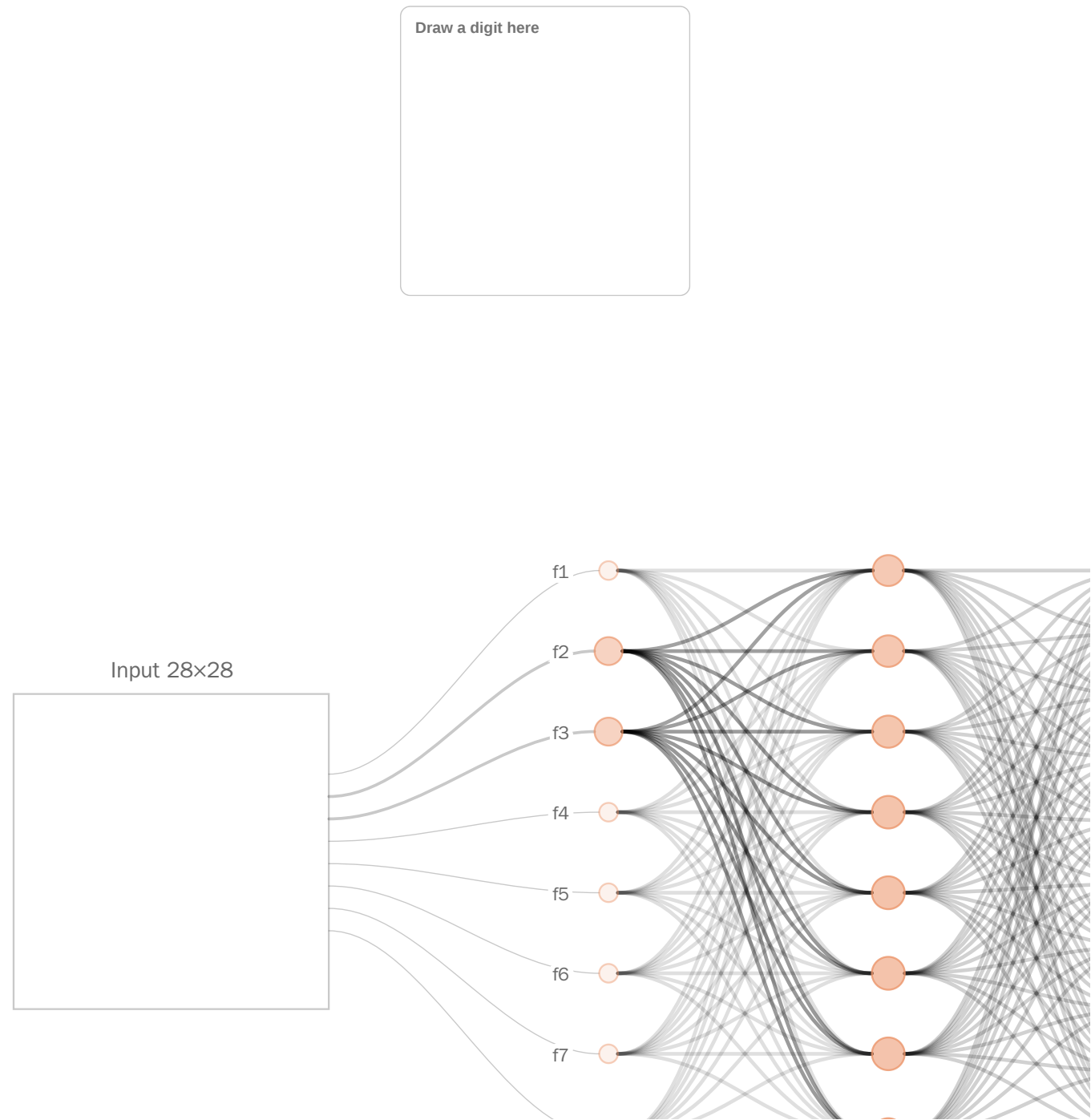


Figure 12 · Interactive MNIST-like neural network. Draw a digit on the left canvas; activations propagate through hidden layers (node size/opacity reflect activation). The right side displays class probabilities (0–9) with the top class emphasized.

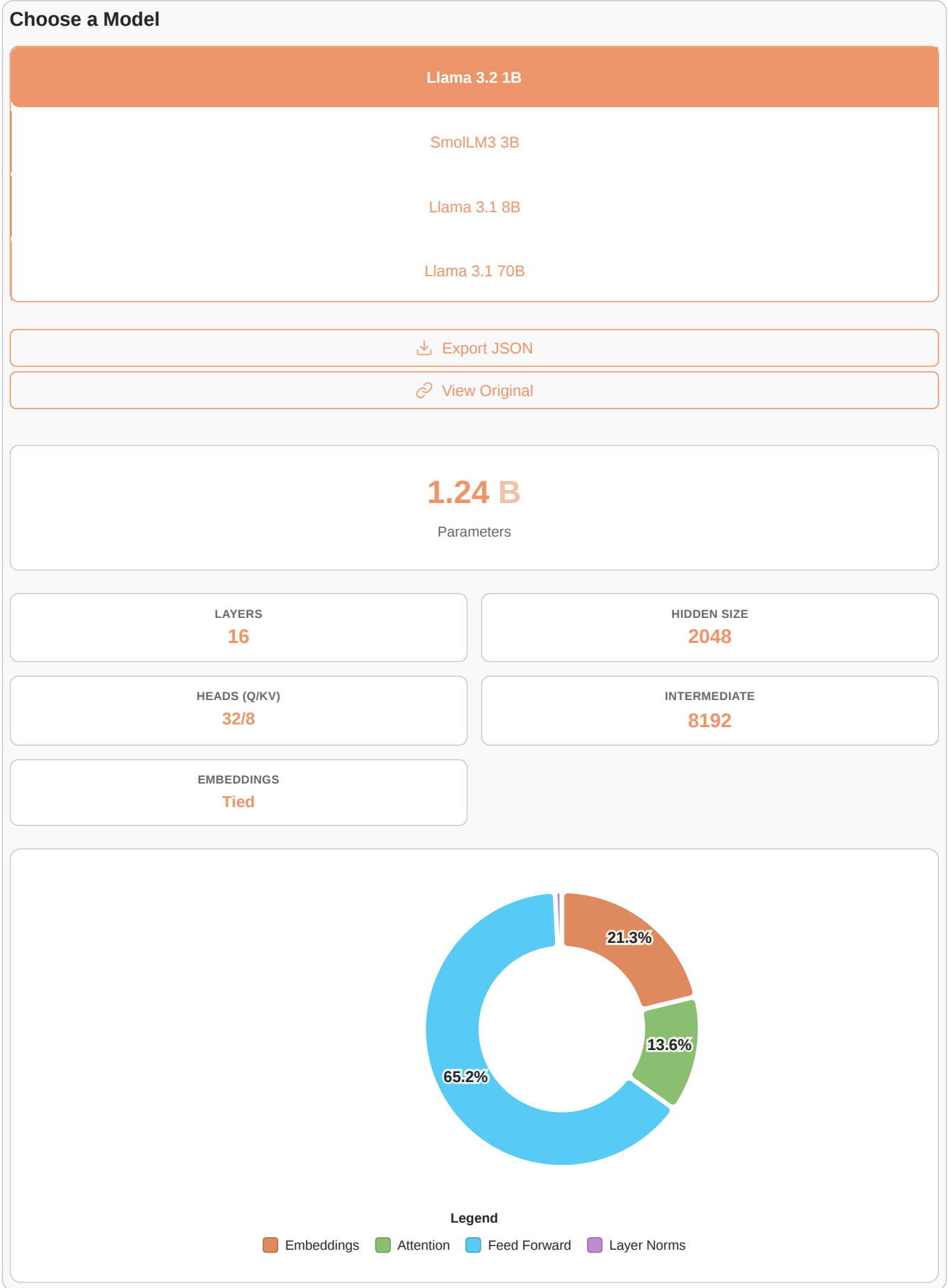
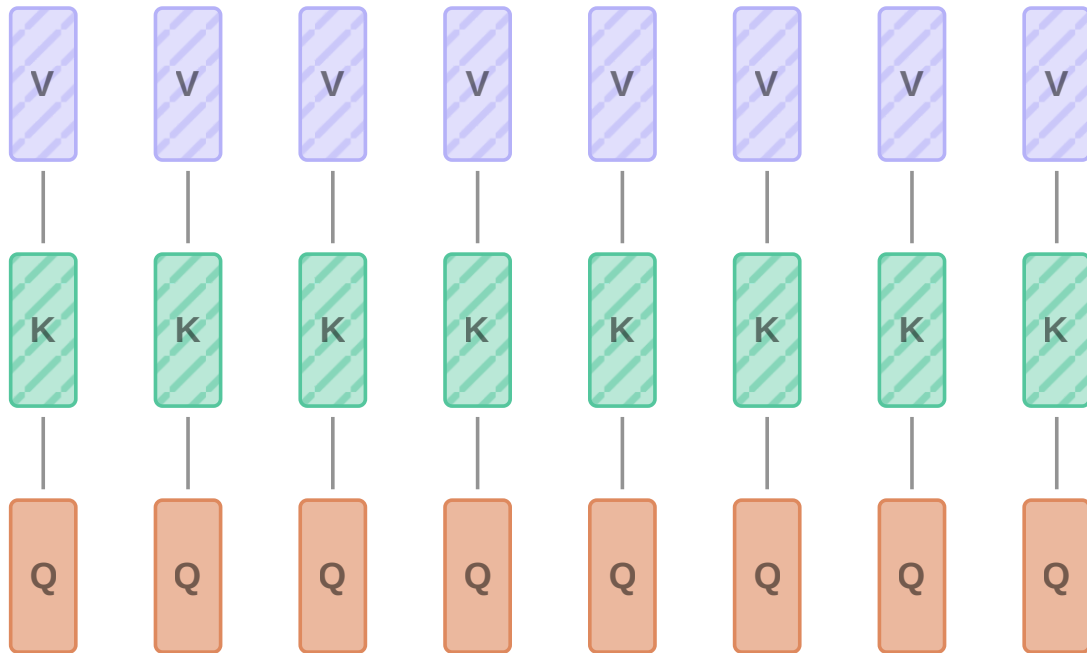


Figure 13 · Visual comparison of parameter distribution between different model sizes (8B vs 70B). Shows how embeddings represent a larger fraction in smaller models.

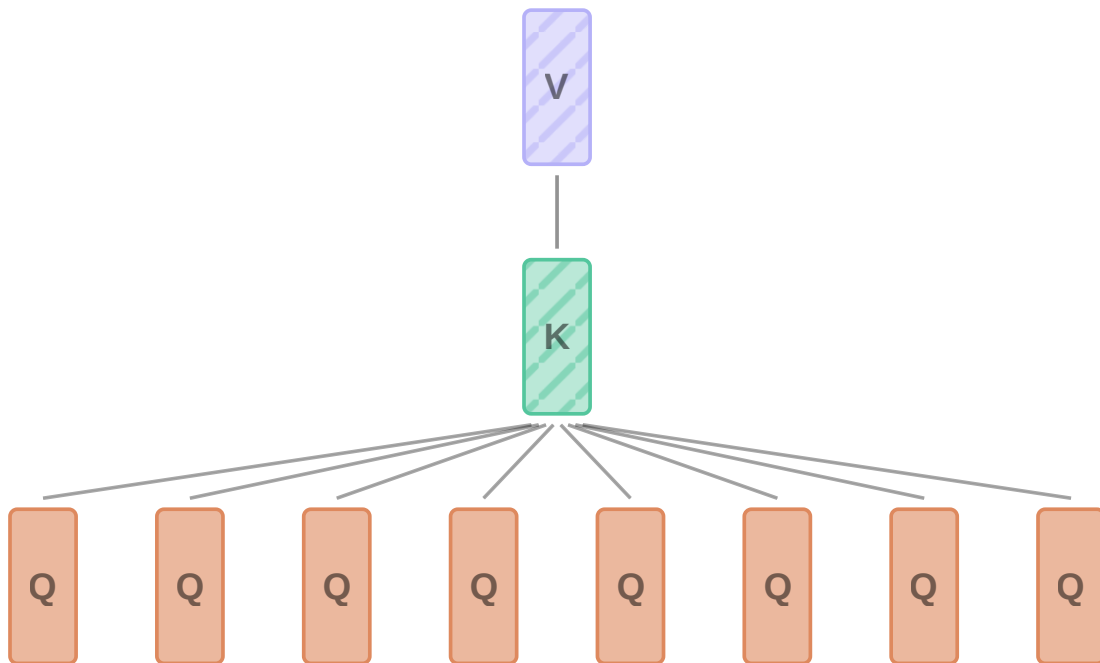
Legend

Values Keys Queries Cached during inference

MULTI HEAD ATTENTION



MULTI QUERY ATTENTION



GROUPED QUERY ATTENTION

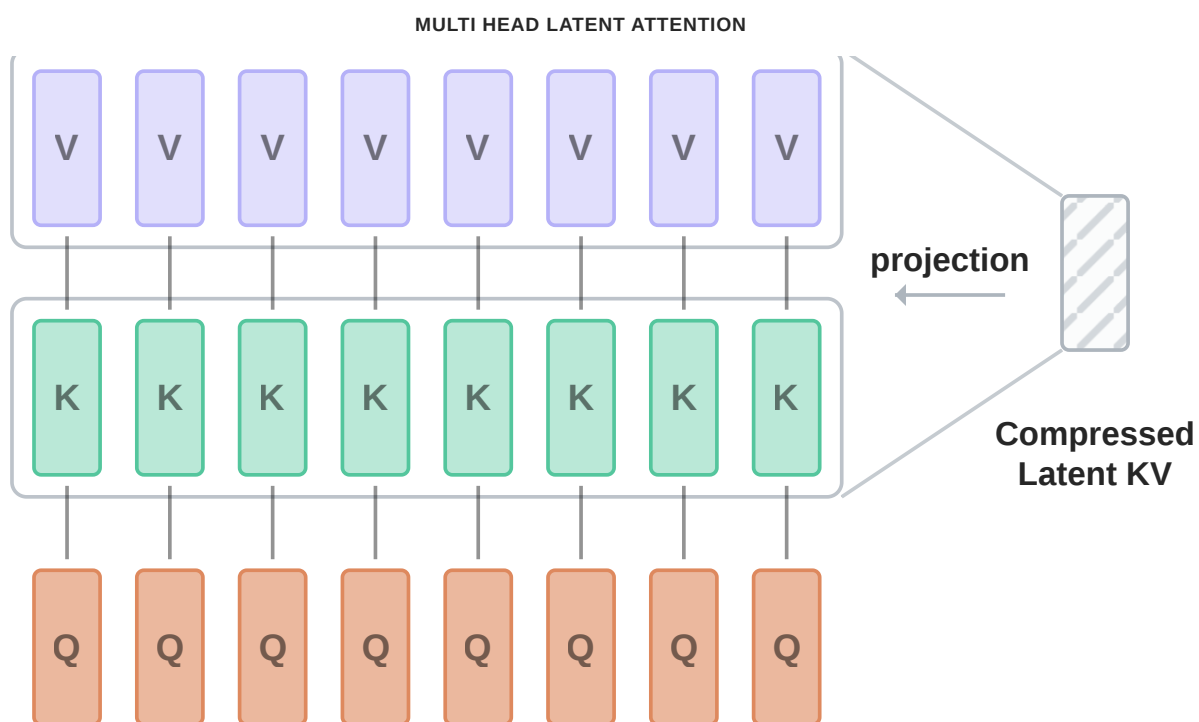
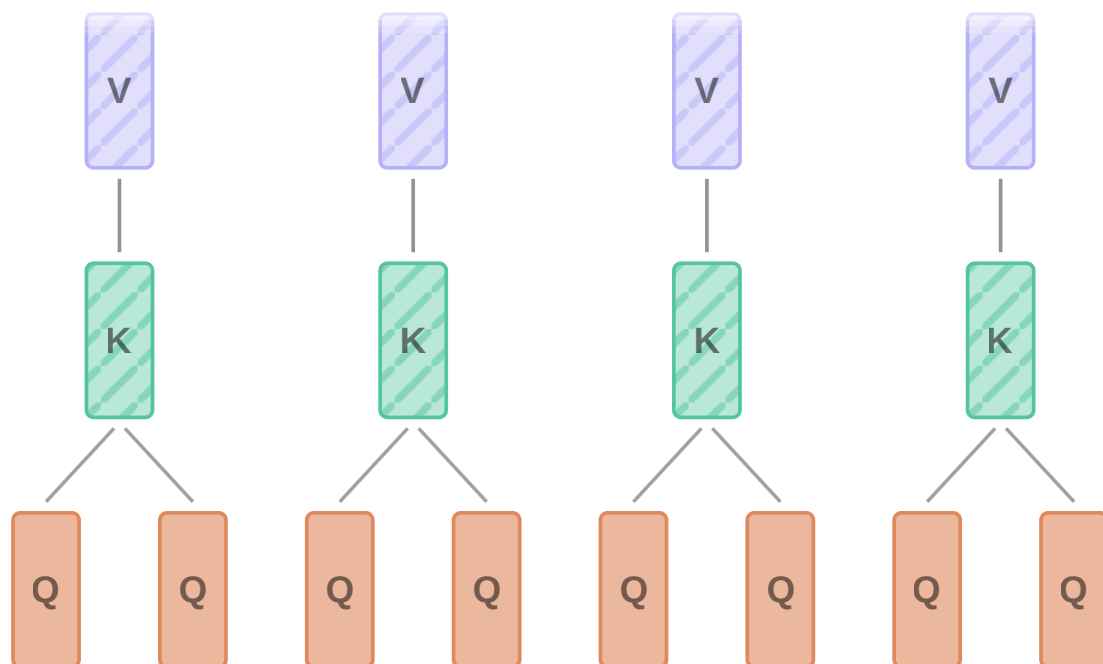


Figure 14 · Simplified illustration of Multi-Head Attention (MHA), Grouped-Query Attention (GQA), Multi-Query Attention (MQA), and Multi-head Latent Attention (MLA). Demonstrates how MLA compresses keys and values into a latent vector.

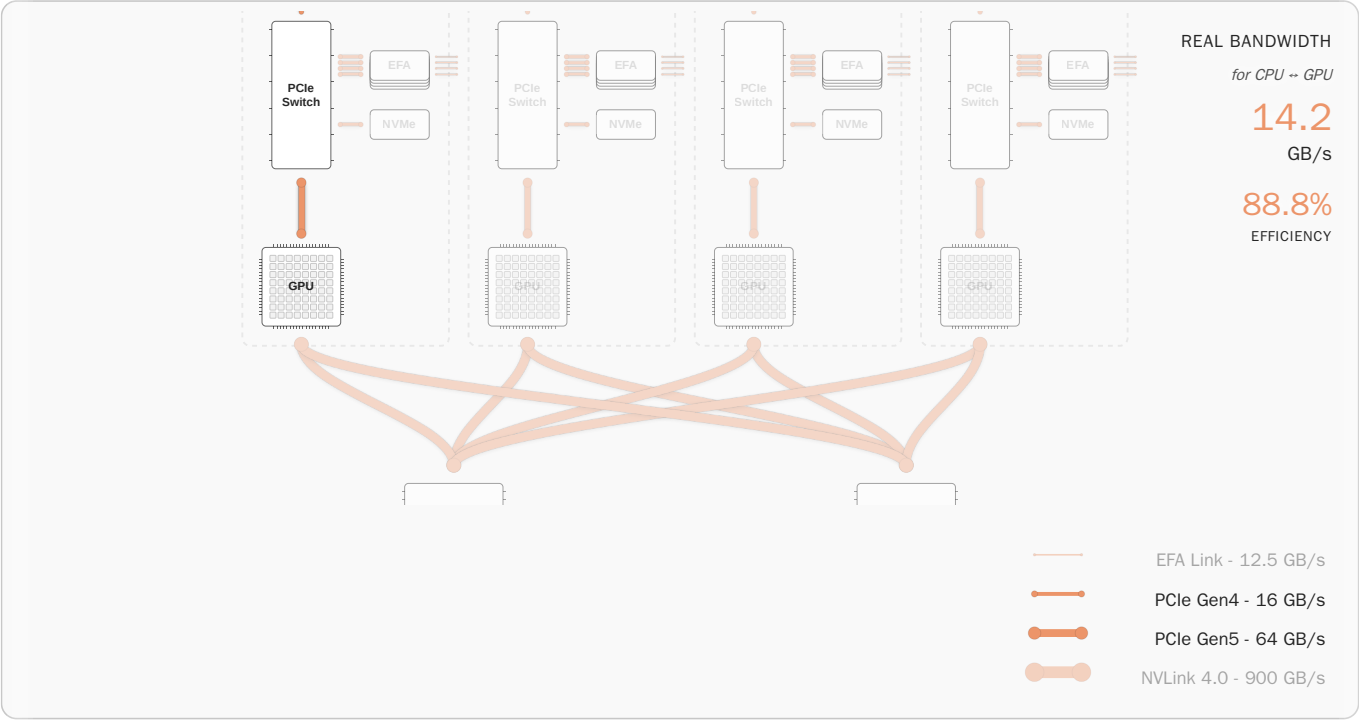


Figure 15 · Interactive diagram of key components and communication links in an AWS p5 instance. Shows the different GPU-GPU communication paths with their actual bandwidths.

[illegible]

	64K	128K	256K	512K	1M	2M	4M	8M
1	0.04	0.06	0.08	0.16	0.31	0.60	1.08	1.08
4	0.04	0.06	0.08	0.16	0.31	0.61	1.08	1.08
8	0.04	0.06	0.08	0.16	0.31	0.73	1.08	1.09
16	0.04	0.06	0.08	0.16	0.31	0.76	1.09	1.09
32	0.04	0.06	0.08	0.16	0.31	0.77	1.09	1.08
64	0.04	0.06	0.08	0.16	0.31	0.74	1.07	1.06

	64K	128K	256K	512K	1M	2M	4M	8M
1	0.14	0.22	0.31	0.39	0.40	0.41	0.51	0.54
4	0.56	0.88	1.19	1.46	1.58	1.81	1.95	2.04
8	1.07	1.66	2.22	2.71	2.89	3.35	3.58	3.73
16	1.98	2.88	3.48	3.92	4.21	4.11	4.15	3.93
32	2.79	3.43	3.81	4.06	4.12	4.04	3.92	3.83
64	3.10	3.79	4.00	3.88	3.89	3.77	3.70	3.67

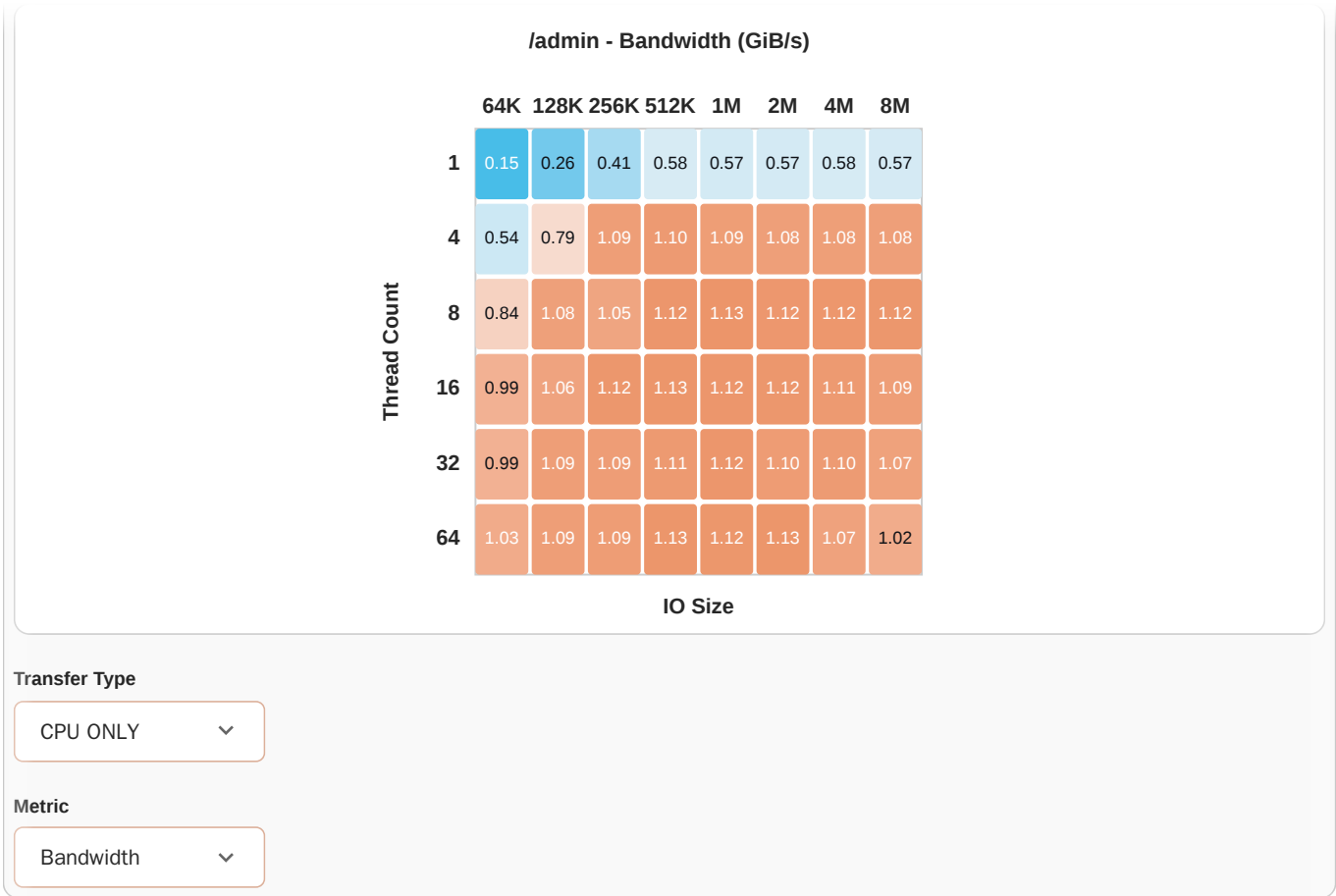


Figure 16 · Interactive heatmaps comparing storage system performance (scratch, root, FSx, admin) across different thread counts and I/O sizes. Visualizes throughput (GiB/s) and IOPS to identify optimal configurations for each storage tier.

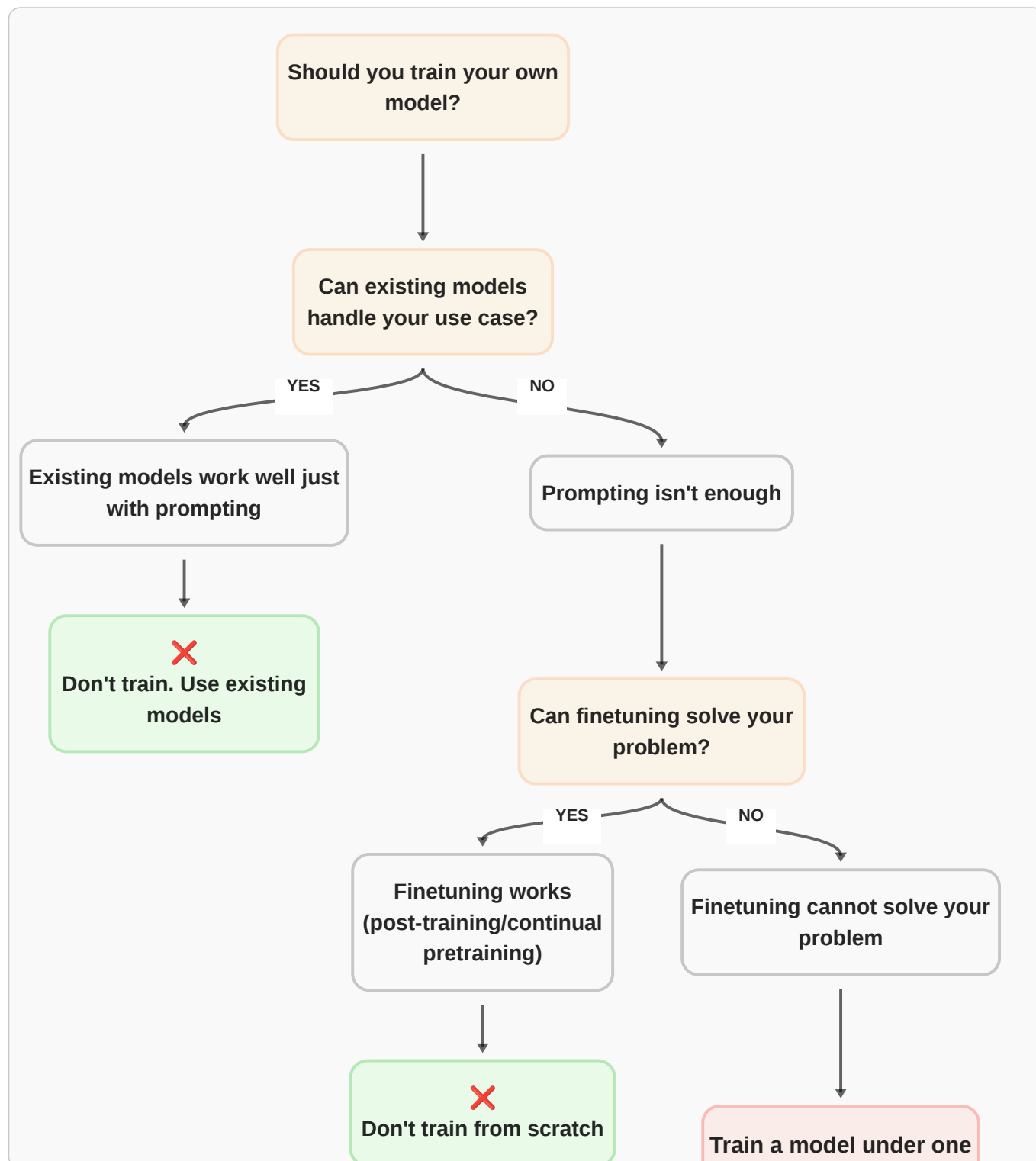


Figure 17 · Interactive decision tree to determine whether you should train your own model. Guides the thought process through key questions: Can existing models handle your use case? Should you fine-tune or train from scratch? For research, production, or strategic open-source?

Generic Line Chart

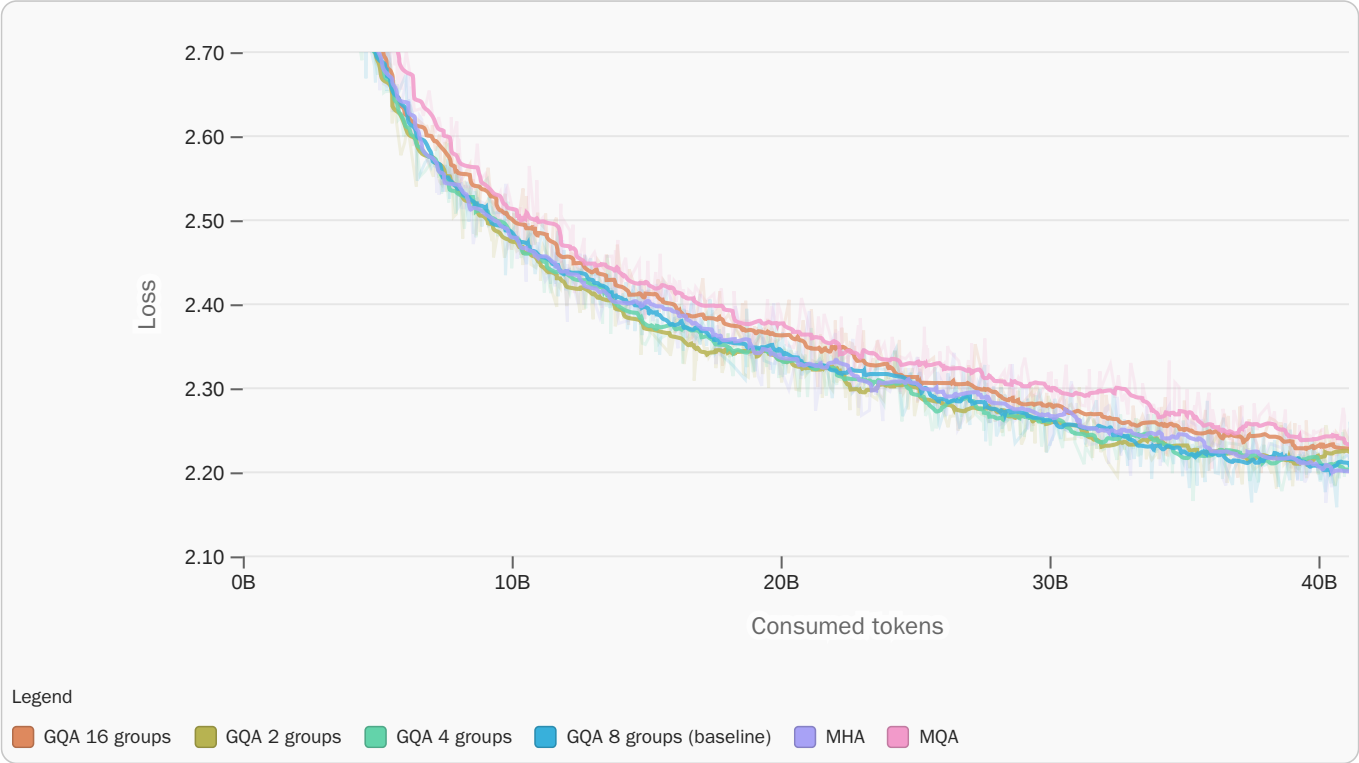


Figure 18 · Configurable line chart with zoom/pan, smoothing and interactive tooltips. Example with loss data for different attention configurations.
Credit: [SmolLM Training Playbook](#)

Generic Six Line Charts Grid

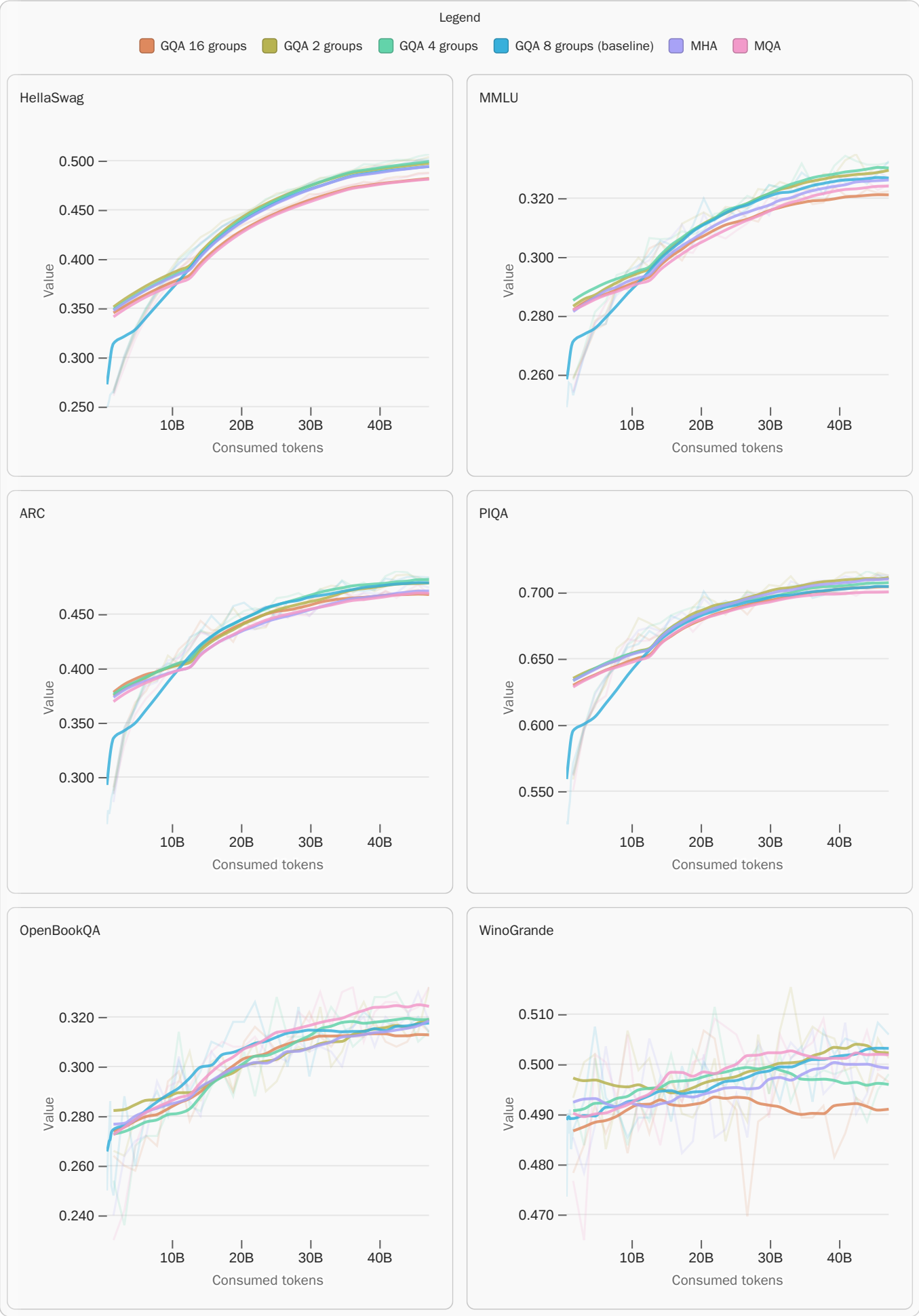


Figure 19 · Configurable grid of line charts with synchronized zoom/pan. Example with evaluations of different metrics.
Credit: [SmolLM Training Playbook](#)

Interactive Mathematical Function Plotter

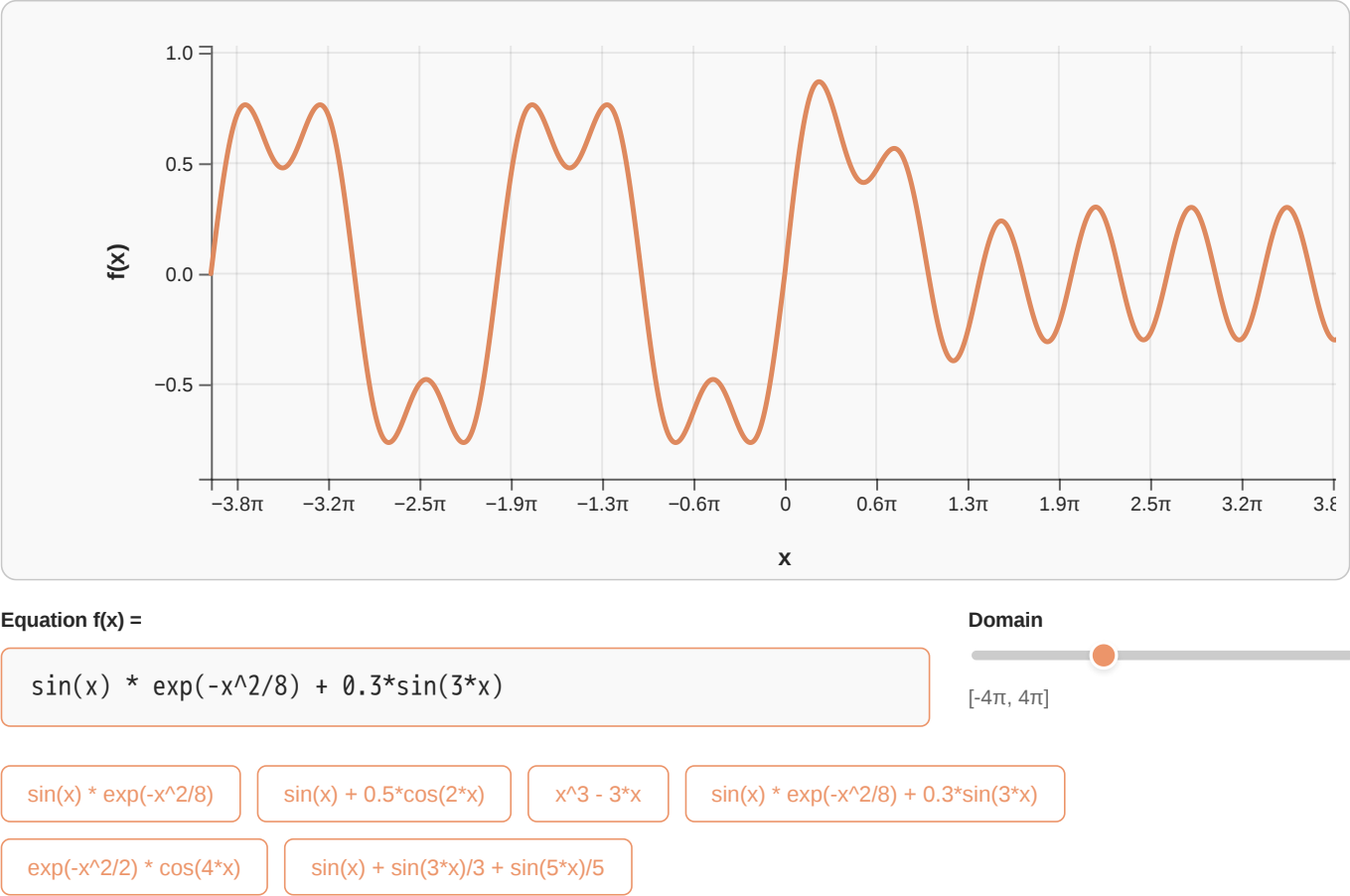


Figure 20 · Interactive equation editor with real-time function plotting. Edit mathematical expressions and see their graphs update instantly. Supports common functions (sin, cos, exp, etc.) with customizable domain range.

Import from LaTeX

⚠ Experimental — May not work with all LaTeX documents.

Transform LaTeX papers into interactive web articles.

Quick Start

```
1 cd app/scripts/latex-importer/
2 cp your-paper.tex input/main.tex
3 cp your-paper.bib input/main.bib
4 node index.mjs
```

What Gets Converted

- `\label{eq:name}` → Interactive equations
- `\ref{eq:name}` → Clickable links
- `\includegraphics{}` → `<Image>` components
- Bibliography integration

Prerequisites

- Pandoc (`brew install pandoc`)
- LaTeX source files and figures

Docker Deployment

Set `ENABLE_LATEX_CONVERSION=true` in your Hugging Face Space to enable automatic conversion during build.

Import from Notion

⚠ Experimental — May not work with all Notion pages.

Convert Notion pages into interactive web articles.

Prerequisites

You need 2 things:

1. `NOTION_TOKEN` — Your Notion integration token
 - Go to notion.so/my-integrations
 - Create a new integration → copy the token (starts with `secret_` or `ntn_`)
2. `NOTION_PAGE_ID` — The ID of your Notion page
 - Open your page in browser: `https://www.notion.so/My-Page-abc123def456`
 - The ID is the last part after the title: `abc123def456`

Share Your Page with the Integration

This step is required! Having a token is not enough — you must explicitly share the page with your integration.

1. Open your Notion page
2. Click "..." (top right corner)
3. Click "Connections" or "Add connections"
4. Select your integration from the list
5. Confirm access

If your page is nested inside another page or database, you can either share the parent page (the integration will have access to all sub-pages) or share each page individually.

Option A: Automatic on HF Space

Add these to your Space settings (Settings → Variables and secrets):

Type	Name	Value
Secret	NOTION_TOKEN	your token
Variable	NOTION_PAGE_ID	your page ID
Variable	ENABLE_NOTION_CONVERSION	true

Then restart the Space. Every rebuild will fetch the latest Notion content automatically.

Option B: Local Development

```
1 cd app/scripts/notion-importer/
2 npm install
3 cp env.example .env
4 # Edit .env with NOTION_TOKEN and NOTION_PAGE_ID
5 npm run notion:import # or: node index.mjs
```

This rewrites `src/content/article.mdx` with your Notion content. Push to deploy.

What Gets Converted

- Images (downloaded locally)
- Callouts → `<Note>` components
- Enhanced tables and code blocks
- Smart link conversion

Writing Tips

Simple guidelines to make your research writing clear, engaging, and effective. Focus on what matters most: helping readers understand your ideas quickly and completely.

Short sections

Break content into small, purpose-driven sections. Each section should answer a single question or support one idea. This improves scanability, helps readers navigate with the TOC, and makes later edits safer.

Clear, minimal annotations

Favor concise captions and callouts that clarify what to look at and why it matters. In code, highlight just the lines that carry the idea; avoid verbose commentary. Precision beats volume.

Explain math notation

Introduce symbols and variables the first time they appear, and prefer well-known identities over custom shorthand. When formulas carry the message, add one sentence of plain-language interpretation right after.

For example, in linear regression with features $x \in \mathbb{R}^d$, weights $w \in \mathbb{R}^d$, and bias b , the prediction is:

$$\hat{y} = w^\top x + b$$

A common training objective is the mean squared error over N samples:

$$\mathcal{L}(w, b) = \frac{1}{N} \sum_{i=1}^N (w^\top x_i + b - y_i)^2$$

Interpretation: the model fits a hyperplane that minimizes the average squared prediction error.

Use the right chart

Picking the right visualization depends on your goal (compare values, show distribution, part-to-whole, trends, relationships, etc.). The Visual Vocabulary poster below provides a concise mapping from analytical task to chart types.



Figure 21 · Credits [Financial-Times](#)
A handy reference to select chart types by purpose — click to enlarge.

Greetings

Huge thanks to the following people for their precious feedbacks!

Leandro von Werra @lvwerra	Clémentine Fourier @clefourrier	Hynek Kydlíček @hynky	Luis Wiedmann @lusxvr
Pablo Montalvo-Leroux @molbap	Lewis Tunstall @lewtun	Guilherme Penedo @guipenedo	Francesco Capuano @fracapitano

Citation

For attribution in academic contexts, please cite this work as

Thibaud Frere (2025). "Bringing paper to life: A modern template for scientific writing".

BibTeX citation

```
@misc{frere2025-bringing-paper-to-life-a-modern-template-for-scientific-writing,
      title={Bringing paper to life: A modern template for scientific writing},
      author={Thibaud Frere},
      year={2025},
      doi={10.1234/abcd.efgh}
}
```

DOI

[10.1234/abcd.efgh](#)

Reuse

Diagrams and text are licensed under [CC-BY 4.0](#) with the source available on [Hugging Face](#), unless noted otherwise. Figures reused from other sources are excluded and marked in their captions (“Figure from …”).

References

- Cover, T. M., & Thomas, J. A. (2006). Entropy, Relative Entropy, and Mutual Information. In *Elements of Information Theory* (2nd ed., pp. 13–55). Wiley.^{[↑](#)}
- Doe, J. (2020). *Learning Efficient Representations for Large-Scale Visual Recognition* [Phdthesis, Massachusetts Institute of Technology]. [10.5555/mit-2020-xyz](#)^{[↑](#)}
- He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778. [10.1109/CVPR.2016.90](#)^{[↑](#)} back: [1](#), [2](#)
- Kingma, D. P., & Ba, J. (2015). Adam: A Method for Stochastic Optimization. *International Conference on Learning Representations (ICLR)*. [https://arxiv.org/abs/1412.6980](#)^{[↑](#)} back: [1](#), [2](#), [3](#)
- McKinney, W. (2017). *Python for Data Analysis* (2nd ed.). O'Reilly Media.^{[↑](#)} back: [1](#), [2](#)
- OpenAI. (2023). *GPT-4 Technical Report* (Techreport arXiv:2303.08774). OpenAI. [https://arxiv.org/abs/2303.08774](#)^{[↑](#)}
- Pedregosa, F., Varoquaux, G., Gramfort, A., & others. (2024). *scikit-learn: Machine Learning in Python (Version 1.4)*. Software. [10.5281/zenodo.592264](#)^{[↑](#)}
- Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., & others. (2020). *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer*. arXiv preprint. [10.48550/arXiv.1910.10683](#)^{[↑](#)}
- Silver, D., Schrittwieser, J., Simonyan, K., Antonoglou, I., Huang, A., & others. (2017). Mastering the game of Go without human knowledge. *Nature*, 550(7676), 354–359. [10.1038/nature24270](#)^{[↑](#)}
- Smith, E., Zhang, W., Rossi, M., & Patel, N. (2024). Privacy-Preserving Training with Low-Precision Secure Aggregation. In A. Smith & B. Johnson (Eds.), *Proceedings of the 41st International Conference on Machine Learning* (Vol. 235, pp. 12345–12367). PMLR. [https://proceedings.mlr.press/v235/](#)^{[↑](#)}
- Smith, J., Lee, A., & Kumar, R. (2021). *ImageNet-21K Subset (Version 2.0)*. Dataset on Zenodo. [10.5281/zenodo.1234567](#)^{[↑](#)}
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Lukasz, & Polosukhin, I. (2017). Attention Is All You Need. *Advances in Neural Information Processing Systems*.^{[↑](#)} back: [1](#), [2](#), [3](#)

Footnotes

- Footnote attached to the sentence above. ^{[↑](#)} back: [1](#), [2](#)
- Multi-paragraph footnote. First paragraph.

Second paragraph with a link to [Astro](#).^{[↑](#)}
- Footnote with a list: ^{[↑](#)}
 - First item
 - Second item
- Footnote with code snippet:

```
1 function add(a: number, b: number) {
2     return a + b;
3 }
```

Result: `add(2, 3) === 5`. [↑](#)

5. Footnote containing citation ([Vaswani et al., 2017](#)) and ([Kingma & Ba, 2015](#)). [↑](#)

6. This footnote includes inline math $E = mc^2$ and a display equation:

$$\int_{-\infty}^{\infty} e^{-x^2} dx = \sqrt{\pi}$$

This is the Gaussian integral, a fundamental result in probability theory. [↑](#)