ComputeCosts observatory report 001
Wikipedia archive, epoch 1

Mateo Mayer

21 February 2026

Abstract

This report documents the first epoch analysis of the Wikipedia archive of the ComputeCosts observatory. The archive records daily pageview activity for a selected set of compute infrastructure related Wikipedia articles and treats these observations as a measurable signal of where practitioners and technically curious readers direct their conceptual attention.

The analysed period runs from 1 January 2025 until 16 February 2026 and contains 8140 observations spanning 407 days across 20 articles. Instead of examining short term fluctuations, the analysis deliberately collapses the entire period into a single robust baseline. Median daily pageviews are used in order to suppress temporary spikes caused by news events or online discussions.

When aggregated into conceptual domains, artificial intelligence related topics dominate the dataset with approximately 88 percent of baseline attention. Within the infrastructure related portion of the dataset, local compute topics account for about 7.3 percent and cloud infrastructure topics account for about 4.4 percent. The resulting cloud to local ratio is approximately 0.5967.

Given the nature of the measurement instrument, this difference should not be interpreted as a quantitative comparison of real world infrastructure adoption. What the dataset does show is that conceptual attention toward cloud and local computing appears in the same order of magnitude within the Wikipedia environment. This indicates that local compute topics form a substantial component of the broader infrastructure discussion rather than a marginal niche.

The analysis therefore treats Wikipedia as a sensor that provides a reproducible signal of conceptual investigation. Its primary value lies in forming one element of a broader observatory in which signals from multiple technical ecosystems can be compared to better understand how practitioners interact with compute infrastructure concepts.

Scope and positioning

This is an independent observatory report. It is not peer reviewed.

The goal is to expose measurable signals, document deterministic methods, and provide auditability through reproducible scripts and exported intermediate files.

The report does not attempt to claim adoption rates of cloud or local systems. It measures conceptual lookup behaviour routed through Wikipedia, and treats this as one observable

component of a broader multi archive evidence stream.

All epoch 1 Wikipedia archive states are immutably anchored on the Algorand blockchain using a dedicated non-fungible token within the COSTS Algorand Standard Asset ecosystem. Each dataset state hash is committed on chain so the archive can be independently verified against a public, tamper evident log. The anchor asset and its transaction history are publicly inspectable at https://explorer.perawallet.app/asset/3456128846/.

Study goal

The goal is to measure whether Wikipedia pageview activity, aggregated robustly over an entire epoch, contains a stable signal that distinguishes attention toward cloud infrastructure concepts versus local compute infrastructure concepts.

A secondary goal is to test whether any local versus cloud conclusion can be plausibly explained by vocabulary bias, where local technical terms naturally route to Wikipedia more than cloud vendor terms.

Dataset

The dataset used in this study consists of daily pageview observations obtained from the Wikimedia pageview service for the English Wikipedia. The archive covers the period from 1 January 2025 until 16 February 2026 and represents the first observation epoch of the ComputeCosts Wikipedia archive.

In total the dataset contains 8140 observations spanning 407 days and covering 20 selected articles related to artificial intelligence, cloud infrastructure, local compute infrastructure and privacy. The relationship between these numbers is straightforward: twenty tracked articles observed over four hundred and seven days produce exactly eight thousand one hundred and forty records. This internal consistency indicates that the archive is complete for the defined article set over the entire epoch.

Each observation represents the number of times a specific Wikipedia article was viewed during a single day. The dataset therefore captures moments in which readers actively seek explanations or context for technical concepts. It should be interpreted as a signal of conceptual investigation rather than a direct measurement of system deployment or operational use.

In other words, the archive does not measure where compute workloads actually run. Instead it measures where practitioners, researchers and technically curious readers direct their attention when they need to understand the underlying ideas.

Method

The objective of the analysis is to obtain a single robust conclusion for the entire observation period rather than a sequence of short term narratives. Internet traffic data are highly sensitive to news cycles, online discussions and temporary hype. If the analysis focuses on weekly or monthly variation, these effects easily dominate the interpretation. For that reason the present study deliberately collapses the full epoch into one baseline representation that reflects the structural behaviour of the dataset.

For every tracked article the baseline attention level is defined as the median of the daily pageview counts observed during the epoch. The median was selected because it suppresses extreme spikes without requiring subjective filtering of events. A single viral day can increase the mean dramatically, but it has almost no effect on the median. Using this statistic therefore produces a stable description of typical attention.

Once a baseline has been obtained for each article, the articles are grouped into a small number of conceptual domains representing different parts of the compute ecosystem. These domains consist of artificial intelligence drivers, cloud infrastructure, local compute infrastructure and privacy related topics. The baseline values of the articles belonging to each domain are summed, after which the relative share of each domain is calculated with respect to the total baseline across all domains.

Because extreme events can still influence results indirectly, the analysis also includes explicit stress tests. The days with the highest total traffic across all tracked articles are removed from the dataset and the entire calculation is repeated. Two scenarios are evaluated: removal of the top one percent of traffic days and removal of the top five percent. If the relative shares remain essentially unchanged, the conclusion can be regarded as robust against short lived spikes.

Finally, the structure of the dataset itself is examined through simple concentration diagnostics. These diagnostics measure how strongly the baseline attention is dominated by a small number of articles. The share of the most dominant article, the combined share of the three most dominant articles and an entropy measure of the baseline distribution are calculated. These quantities help interpret whether the infrastructure signals appear independently or are largely driven by a broader wave of interest surrounding artificial intelligence.

Results

The robust baseline derived from the median daily pageviews provides a stable picture of how attention is distributed across the tracked concepts during the full epoch. When the article baselines are aggregated into the four conceptual domains defined earlier, artificial intelligence dominates the landscape with a baseline share of approximately 0.882831. Local compute concepts account for about 0.073156 of the attention, cloud infrastructure concepts account for roughly 0.043650, and privacy related topics represent only a very small fraction at approximately 0.000364.

These values indicate that the overall conceptual environment in which compute infrastructure is discussed is strongly shaped by the current artificial intelligence wave. Infrastructure questions appear as a secondary layer beneath that broader context. Within that layer, however, local compute topics receive noticeably more attention than cloud infrastructure topics.

The relationship becomes clearer when the two are compared directly. The ratio between the cloud and local baselines is approximately 0.5967. In practical terms this means that local compute topics receive about one point seven times the structural attention of cloud infrastructure topics within this dataset. Because the baseline is derived from the median of the entire observation period, this relationship reflects typical behaviour rather than a few highly active days.

To verify that the result is not driven by temporary spikes, the analysis removes the most extreme traffic days and recomputes the shares. Removing the top one percent of days produces almost

identical values, and removing the top five percent of days leads to only negligible changes. The cloud versus local relationship therefore appears to be stable and not dependent on short lived bursts of activity.

Additional diagnostics confirm this stability. The ten highest traffic days together account for roughly 6.856 percent of the total view volume, meaning that more than ninety three percent of all observations originate from ordinary days rather than exceptional events. The dataset therefore behaves more like a steady attention stream than a sequence of viral spikes.

The structure of the baseline distribution also reveals that attention is strongly concentrated. The single most viewed article accounts for approximately 0.636 of the baseline, while the three most dominant articles together account for about 0.813. This concentration indicates that the dataset is largely shaped by a small number of highly visible topics, most of which belong to the artificial intelligence domain. As a result the cloud versus local comparison should be interpreted as a secondary signal embedded within a much larger AI driven attention pattern.

The numerical difference between cloud and local attention should not be interpreted as a precise quantitative comparison of real world infrastructure adoption. The measurement instrument in this report is Wikipedia pageview activity, which reflects conceptual lookup behaviour rather than operational deployment. What the data do show reliably is that attention directed toward cloud infrastructure concepts and attention directed toward local compute concepts appear in the same order of magnitude within this environment. Even when the analysis suppresses short term spikes and aggregates the entire epoch into a single robust baseline, both domains remain clearly visible. This indicates that local compute topics form a substantial component of the broader infrastructure discussion rather than a marginal niche.

Critical analysis

Although the numerical results are internally consistent, their interpretation requires caution. The dataset measures conceptual lookup behaviour routed through Wikipedia. It does not directly measure infrastructure deployment, operational costs or real world adoption decisions. The archive should therefore be understood as an attention sensor rather than a ground truth indicator.

One important limitation arises from vocabulary asymmetry. Many local compute concepts such as graphics processing units, CUDA or data centres are generic technical terms for which Wikipedia naturally serves as an explanatory resource. Cloud infrastructure, in contrast, is often approached through vendor documentation, pricing calculators or management consoles. These pathways may bypass Wikipedia entirely. As a result the platform may structurally amplify certain kinds of curiosity while underrepresenting others.

A second factor is the dominance of artificial intelligence in the current technological discourse. Curiosity about large language models frequently leads readers toward hardware topics such as GPUs or inference infrastructure. Some of the observed attention directed toward local compute concepts may therefore reflect the AI boom rather than deliberate evaluation of infrastructure strategies.

Despite these limitations the signal remains informative. The baseline is robust against spikes, the method is deterministic, and the results can be reproduced from publicly available data. Because

the same procedure can be repeated in later epochs and across other archives such as GitHub, Stack Overflow, Hacker News and Reddit, the observatory can gradually build a multi signal view of how practitioners interact with compute infrastructure concepts.

For future iterations the analysis could be strengthened by expanding the cloud related concept set. Including articles such as infrastructure as a service, platform as a service, serverless computing, virtual machines, containerisation and Kubernetes would help balance the conceptual vocabulary and reduce the risk that the measurement favours hardware oriented terminology. A matched concept comparison between cloud and local paradigms would also make the inference more symmetrical.

For this reason the Wikipedia archive should be interpreted as one sensor within a larger measurement framework rather than as a definitive source of evidence on its own. Different technical ecosystems route curiosity in different ways. Some questions lead naturally to encyclopedic resources, while others are resolved through documentation, code repositories or discussion forums. The value of the present archive lies in providing a transparent and reproducible signal that can later be compared with observations from other archives such as GitHub, Stack Overflow, Hacker News and Reddit. When analysed together, these independent streams can reveal patterns that are unlikely to be explained by the biases of any single platform.

Conclusions

Wikipedia pageviews can provide a stable and reproducible attention signal when aggregated over an entire epoch using robust statistics. The present analysis shows that, within this dataset, conceptual attention toward cloud and local computing appears in the same order of magnitude. This indicates that local compute topics represent a substantial component of the broader infrastructure conversation rather than a marginal niche. Because Wikipedia reflects conceptual lookup behaviour, the archive should be interpreted as one sensor within a larger observatory rather than as a direct measurement of infrastructure adoption. The value of the approach lies in combining this signal with observations from other archives to build a broader empirical picture of compute friction and deployment choices.

Appendix A, Key numeric outputs from epoch 1

Epoch rows, 8140.

Epoch date range, 2025-01-01 to 2026-02-16.

Articles, 20.

Days, 407.

Baseline shares using median daily views.

Artificial intelligence, 0.882831.

Local compute, 0.073156.

Cloud infrastructure, 0.043650.

Privacy, 0.000364.

Cloud to local ratio, 0.5967.

Top 10 days fraction of all views, 0.06856.

Baseline concentration.

Top 1 share, 0.636091.

Top 3 share, 0.813116.

Entropy, 1.456062.

Appendix B, Python analysis script

```
#!/usr/bin/env python3
import json
from dataclasses import dataclass
from pathlib import Path
import numpy as np
import pandas as pd


INPUT_PATH = Path("/mnt/data/wikipedia.jsonl")

EPOCH_START = "2025-01-01"
EPOCH_END = "2026-02-28"


CLOUD_ARTICLES = [
"Cloud_computing",
"Amazon_Web_Services",
"Microsoft_Azure",
"Google_Cloud_Platform",
"Vendor_lock-in",
]

LOCAL_ARTICLES = [
"On-premises_software",
"Edge_computing",
"Data_center",
"Graphics_processing_unit",
```

```
"CUDA",
"Nvidia",
]

PRIVACY_ARTICLES = [
"Data_privacy",
]

AI_DRIVER_ARTICLES = [
"Artificial_intelligence",
"Machine_learning",
"Deep_learning",
"Large_language_model",
"Transformer_(machine_learning_model)",
"Inference",
"OpenAI",
"ChatGPT",
]

GROUPS = {
        "cloud": CLOUD_ARTICLES,
        "local": LOCAL_ARTICLES,
        "privacy": PRIVACY_ARTICLES,
        "ai": AI_DRIVER_ARTICLES,
}


@dataclass(frozen=True)
class Outputs:
article_epoch_summary_csv: Path
group_epoch_summary_csv: Path
robustness_checks_csv: Path


def load_jsonl(path: Path) -> pd.DataFrame:
records = []
with path.open("r", encoding="utf-8") as f:
for line_no, line in enumerate(f, start=1):
line = line.strip()
if not line:
continue
try:
records.append(json.loads(line))
except json.JSONDecodeError as e:
raise ValueError(f"Invalid JSON at line {line_no}: {e}") from e

df = pd.DataFrame(records)
```

```
required = {"article", "timestamp", "views", "project", "source"}
missing = required - set(df.columns)
if missing:
raise ValueError(f"Missing required fields in JSONL: {sorted(missing)}")

df["date"] = pd.to_datetime(df["timestamp"].astype(str).str.slice(0, 8),
     format="%Y%m%d", errors="raise")
df["views"] = pd.to_numeric(df["views"], errors="raise")

projects = sorted(df["project"].dropna().unique().tolist())
sources = sorted(df["source"].dropna().unique().tolist())
if projects != ["en.wikipedia.org"]:
raise ValueError(f"Unexpected project values: {projects}")
if sources != ["wikimedia_pageviews"]:
raise ValueError(f"Unexpected source values: {sources}")

return df


def restrict_epoch(df: pd.DataFrame, start: str, end: str) -> pd.
    DataFrame:
start_ts = pd.Timestamp(start)
end_ts = pd.Timestamp(end)
out = df[(df["date"] >= start_ts) & (df["date"] <= end_ts)].copy()
if out.empty:
raise ValueError("No rows left after epoch restriction, check epoch
    dates or input file.")
return out


def daily_matrix(df_epoch: pd.DataFrame) -> pd.DataFrame:
daily = (
df_epoch.pivot_table(index="date", columns="article", values="views",
    aggfunc="sum")
.fillna(0)
.sort_index()
)
return daily


def trimmed_mean(x: np.ndarray, trim_frac: float) -> float:
if len(x) == 0:
return float("nan")
if trim_frac <= 0:
return float(np.mean(x))
x_sorted = np.sort(x)
n = len(x_sorted)
k = int(np.floor(n * trim_frac))
```

```
if 2 * k >= n:
return float(np.median(x_sorted))
return float(np.mean(x_sorted[k : n - k]))


def winsorized_mean(x: np.ndarray, winsor_frac: float) -> float:
if len(x) == 0:
return float("nan")
if winsor_frac <= 0:
return float(np.mean(x))
x_sorted = np.sort(x)
n = len(x_sorted)
k = int(np.floor(n * winsor_frac))
if k == 0:
return float(np.mean(x_sorted))
lo = x_sorted[k]
hi = x_sorted[n - k - 1]
x_w = np.clip(x_sorted, lo, hi)
return float(np.mean(x_w))


def compute_article_baselines(daily: pd.DataFrame, trim_frac: float =
    0.10, winsor_frac: float = 0.05) -> pd.DataFrame:
rows = []
for article in daily.columns:
x = daily[article].to_numpy(dtype=float)
rows.append(
{
        "article": article,
        "total_views": float(np.sum(x)),
        "mean_daily": float(np.mean(x)),
        "median_daily": float(np.median(x)),
        "trimmed_mean_daily_10pct": trimmed_mean(x, trim_frac),
        "winsor_mean_daily_5pct": winsorized_mean(x, winsor_frac),
        "max_daily": float(np.max(x)),
        "p95_daily": float(np.quantile(x, 0.95)),
}
)
df = pd.DataFrame(rows).sort_values("total_views", ascending=False)
df["spike_ratio_max_over_p95"] = df["max_daily"] / df["p95_daily"].
    replace(0, np.nan)
return df


def compute_group_baseline_shares(article_baselines: pd.DataFrame,
    baseline_field: str) -> pd.DataFrame:
baseline_map = article_baselines.set_index("article")[baseline_field].
    to_dict()
```

```
def sum_group(articles):
missing = [a for a in articles if a not in baseline_map]
if missing:
raise ValueError(f"Missing expected articles in dataset: {missing}")
return float(sum(baseline_map[a] for a in articles))

group_vals = {g: sum_group(arts) for g, arts in GROUPS.items()}
total = float(sum(group_vals.values()))
out = []
for g, v in group_vals.items():
out.append(
{
       "group": g,
       "baseline_daily": v,
       "baseline_share": (v / total) if total > 0 else float("nan"),
}
)

df = pd.DataFrame(out).sort_values("baseline_share", ascending=False)
df["baseline_field"] = baseline_field
df["cloud_to_local_ratio"] = float(group_vals["cloud"] / group_vals["
    local"]) if group_vals["local"] > 0 else float("inf")
return df


def remove_top_days_and_recompute(daily: pd.DataFrame, baseline_field:
    str, remove_top_frac: float) -> pd.DataFrame:
totals = daily.sum(axis=1)
n = len(totals)
k = int(np.ceil(n * remove_top_frac))
if k <= 0:
raise ValueError("remove_top_frac produced k <= 0")

drop_idx = totals.sort_values(ascending=False).head(k).index
daily_cut = daily.drop(index=drop_idx)

art = compute_article_baselines(daily_cut)
grp = compute_group_baseline_shares(art, baseline_field)
grp = grp[["group", "baseline_share"]].copy()
grp["remove_top_frac_days"] = remove_top_frac
grp["days_removed"] = k
grp["days_remaining"] = len(daily_cut)
return grp


def spike_contribution(daily: pd.DataFrame, top_days: int = 10) -> pd.
    DataFrame:
```

```
totals = daily.sum(axis=1).sort_values(ascending=False)
top_sum = float(totals.head(top_days).sum())
all_sum = float(totals.sum())
frac = top_sum / all_sum if all_sum > 0 else float("nan")
return pd.DataFrame(
[
{
       "metric": "total_all_articles",
       "top_days": top_days,
       "top_days_views": top_sum,
       "all_views": all_sum,
       "fraction_from_top_days": frac,
}
]
)


def concentration_on_baseline(article_baselines: pd.DataFrame,
    baseline_field: str) -> pd.DataFrame:
x = article_baselines[baseline_field].to_numpy(dtype=float)
x = np.maximum(x, 0.0)
s = float(np.sum(x))
if s <= 0:
return pd.DataFrame(
[{"baseline_field": baseline_field, "top1_share": float("nan"), "
    top3_share": float("nan"), "entropy": float("nan")}]
)

p = x / s
top1 = float(np.sort(p)[-1])
top3 = float(np.sort(p)[-3:].sum()) if len(p) >= 3 else float(np.sum(p))
entropy = float(-np.sum(p * np.log(p + 1e-15)))
return pd.DataFrame([{"baseline_field": baseline_field, "top1_share":
    top1, "top3_share": top3, "entropy": entropy}])


def main() -> None:
df = load_jsonl(INPUT_PATH)
df_epoch = restrict_epoch(df, EPOCH_START, EPOCH_END)
daily = daily_matrix(df_epoch)

print(f"Epoch rows: {len(df_epoch)}")
print(f"Epoch date range: {daily.index.min().date()} to {daily.index.max
    ().date()}")
print(f"Articles: {daily.shape[1]}, Days: {daily.shape[0]}")

article_baselines = compute_article_baselines(daily)
```

```
baseline_field = "median_daily"
group_summary = compute_group_baseline_shares(article_baselines,
    baseline_field)

r1 = remove_top_days_and_recompute(daily, baseline_field,
    remove_top_frac=0.01)
r5 = remove_top_days_and_recompute(daily, baseline_field,
    remove_top_frac=0.05)
robust_checks = pd.concat([r1, r5], ignore_index=True)

spike_diag = spike_contribution(daily, top_days=10)
conc_diag = concentration_on_baseline(article_baselines, baseline_field)

print("\nPeriod level conclusion, robust baseline shares using median
    daily views:")
print(group_summary[["group", "baseline_daily", "baseline_share"]].
    to_string(index=False))

print("\nRobustness check, baseline shares after removing highest
    traffic days:")
print(robust_checks.to_string(index=False))

print("\nSpike contribution diagnostic:")
print(spike_diag.to_string(index=False))

print("\nBaseline concentration diagnostic:")
print(conc_diag.to_string(index=False))


if __name__ == "__main__":
main()
```

Appendix C, Archive verification

All records used in this study are part of the epoch 1 archive whose state hashes are anchored on the Algorand blockchain through the COSTS asset. The public explorer entry allows independent verification of the committed dataset state.

The anchor NFT asset and its transaction history are publicly inspectable at https://explorer.perawallet.app/asset/3456128846/.