

Agentic AI Demo Report

Table of contents

Overview	1
Cleaning Summary	2
RT Histogram (kept trials)	2
Model Metrics	3
Frequency Effect on Recognition Speed	4
Frequency and response time	5

```
library(yaml)
library(readr)
library(dplyr)

metrics_path <- here::here("outputs", "results", "metrics.yml")
cleaning_path <- here::here("outputs", "results", "cleaning.yml")
fig_path <- here::here("outputs", "figures", "rt_hist.png")
freq_summary_path <- here::here("outputs", "data", "character_frequency_rt_summary.csv")
freq_curve_path <- here::here("outputs", "data", "frequency_rt_curve.csv")
freq_summary_text_path <- here::here("outputs", "results", "frequency_rt_summary.md")
freq_fig_path <- here::here("outputs", "figures", "frequency_vs_rt.png")

metrics <- yaml::read_yaml(metrics_path)
cleaning <- yaml::read_yaml(cleaning_path)
N <- as.integer(metrics$n_obs)

# helpers for formatting
fmt3 <- function(x) sprintf("%.3f", x)
fmt6 <- function(x) sprintf("%.6f", x)
```

Overview

This report reads pre-computed outputs from the simple demo pipeline.

- Processed data: `outputs/data/processed.csv`
- Cleaning summary: `outputs/results/cleaning.yml`

- Model metrics: `outputs/results/metrics.yml`

Cleaning Summary

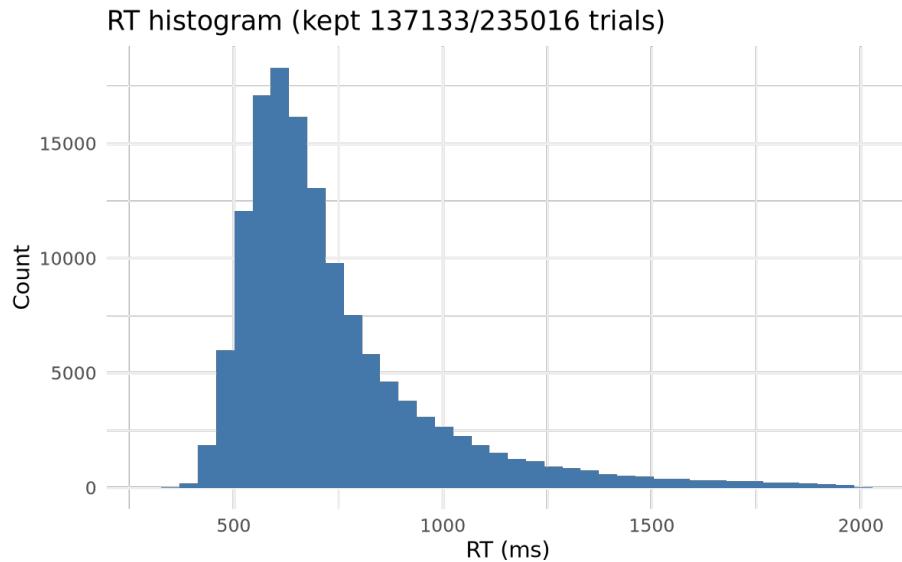
The pipeline kept 137133 of 235016 trials (dropped 97883). Settings: correct-only = TRUE, RT range = 200–2000 ms.

```
data.frame(
  setting = c("correct_only", "rt_min_ms", "rt_max_ms", "total_trials", "kept_trials", "dropped_trials"),
  value = c(
    as.character(cleaning$trimming$correct_only),
    cleaning$trimming$rt_min_ms,
    cleaning$trimming$rt_max_ms,
    cleaning$counts$total_trials,
    cleaning$counts$kept_trials,
    cleaning$counts$dropped_trials
  )
)

      setting   value
1  correct_only    TRUE
2       rt_min_ms     200
3       rt_max_ms    2000
4  total_trials  235016
5   kept_trials  137133
6 dropped_trials  97883
```

RT Histogram (kept trials)

```
knitr::include_graphics(fig_path)
```



Model Metrics

Model: lm(mean_log_rt ~ log_freq + strokes) (N = 3852)

$R^2 = 0.434$.

```
cat(paste0("Adjusted R² = ", fmt3(as.numeric(metrics$adj_r2)), ".\n\n"))

Adjusted R² = 0.433.

cat(paste0("Residual sigma = ", fmt3(as.numeric(metrics$sigma)), ".\n\n"))

Residual sigma = 0.099.

cat("Information criteria:\n\n")

Information criteria:

print(data.frame(
  metric = c("AIC", "BIC"),
  value = c(fmt3(as.numeric(metrics$aic)), fmt3(as.numeric(metrics$bic)))
))

  metric      value
1    AIC -6851.160
2    BIC -6826.134

Coefficients:

data.frame(
  term = c("intercept", "log_freq", "strokes"),
```

```

estimate = c(
  fmt6(as.numeric(metrics$coefficients$intercept)),
  fmt6(as.numeric(metrics$coefficients$log_freq)),
  fmt6(as.numeric(metrics$coefficients$strokes))
)
)

term estimate
1 intercept 6.452355
2 log_freq -0.070823
3 strokes 0.013355

```

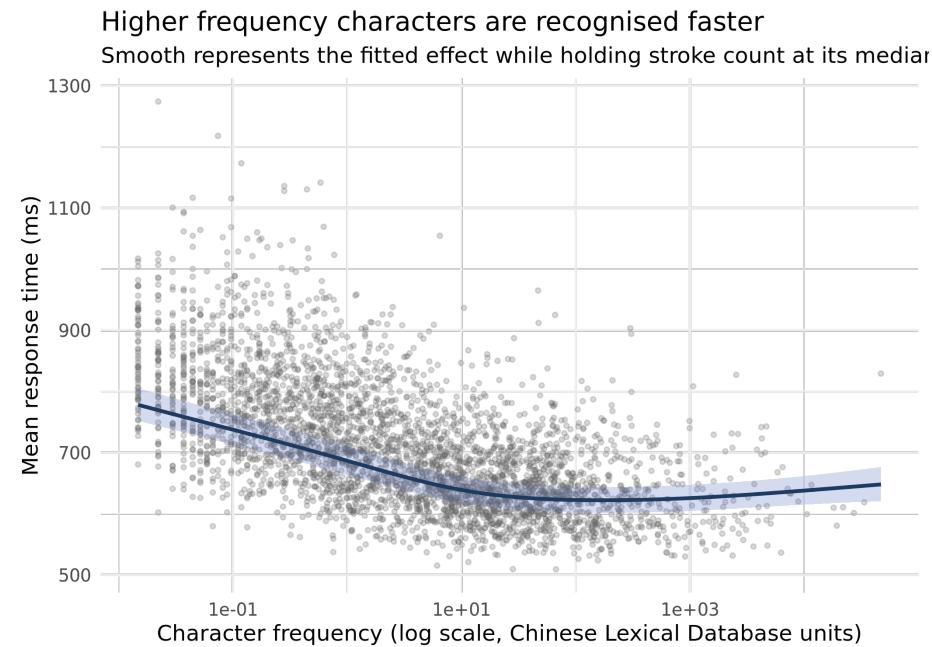
Frequency Effect on Recognition Speed

```

character_frequency <- readr::read_csv(freq_summary_path, show_col_types = FALSE)
frequency_curve <- readr::read_csv(freq_curve_path, show_col_types = FALSE)

knitr::include_graphics(freq_fig_path)

```



```

character_frequency |>
  mutate(decile = ntile(frequency, 10)) |>
  group_by(decile) |>
  summarise(
    freq_range = sprintf(

```

```

    "%s-%s",
    scales::comma(round(min(frequency), 1)),
    scales::comma(round(max(frequency), 1))
),
mean_rt_ms = round(mean(mean_rt_ms), 1),
.groups = "drop"
)

# A tibble: 10 x 3
decile freq_range mean_rt_ms
<int> <chr>          <dbl>
1      1 0-0            821
2      2 0-0            792.
3      3 0-1            768.
4      4 1-1            729.
5      5 1-3            711.
6      6 3-7            680.
7      7 7-18           654.
8      8 18-42           641.
9      9 42-126          645.
10     10 127-46,944     628.

cat(readr::read_file(freq_summary_text_path))

```

Frequency and response time

- Trials analysed: 99138 (subjects: 29; characters: 3852)
- Median stroke count held constant in effect estimates: 9

Frequency effect (holding stroke count at the median):

- Moving from the 10th to 50th percentile of frequency is associated with a 83.7 ms faster response.
- Moving from the 50th to 90th percentile yields a further 32.3 ms reduction.
- Between the 90th and 99th percentile the model shows only a 8.4 ms increase, indicating a flat or slightly reversing trend at the very high end.

Overall, characters in the 99th percentile respond about 14.6% faster than those in the 10th percentile.