# Main Paper Analysis

2025-06-27

## Contents



This is the main analysis code for all models reported in the main paper, including the generation of effect sizes and equivalence testing. Please ensure you have the data frames 'mixedeffect_df.csv' and 'betweenmeasures_df.csv' downloaded from the OSF project. This code will look for these files in a folder named 'OSF_data'.

Alternatively, you can download 'main_data.csv' and use the 'main_datawrangle.Rmd' document to wrangle the data yourself, or to check the data processing code.

A knitted version of this file is available in the OSF project folder under 'main_analysis_processed.pdf'.

### 0.1 Reading in the data files

This will use the here package.

```
mixedeffect_df <- read.csv(here("OSF_data", "mixedeffect_df.csv"))
betweenmeasure_df <- read.csv(here("OSF_data", "betweenmeasures_df.csv"))
```

### 0.2 Randomisation checks

Below shows the successful randomisation across the training conditions (between measures).

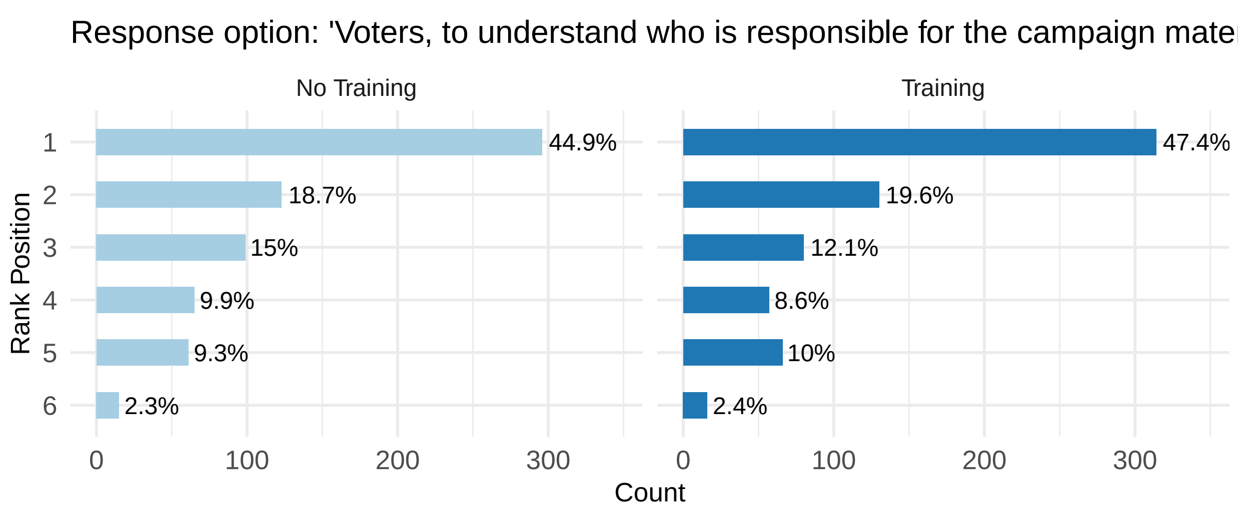
Table 1: Randomisation Check Across Demographics

| | Test | Statistic.t | p_value |
|---|---|---|---|
| Age | t-test | 0.1 | 0.916 |
| Political interest | t-test | 0.19 | 0.852 |
| Gender | Chi-squared | 0.36 | 0.948 |
| Education | Chi-squared | 1.99 | 0.851 |
| Party ID | Chi-squared | 6.88 | 0.865 |
| Ethnicity | Chi-squared | 4.8 | 0.308 |

## 0.3 Manipulation check

Below shows where in the ranking participants tended to rate 'voters' when asked who the digital imprint information was most useful for.

```
## `summarise()` has grouped output by 'Training.condition'. You can override
## using the `.groups` argument.
```




```
# Wilcoxon signed-rank test for overall distribution of rank

betweenmeasure_df$useful_rank_1_num <- as.numeric(as.character(betweenmeasure_df$useful_rank_1))

wilcox.test(useful_rank_1_num ~ Training.condition, data = betweenmeasure_df,
            alternative = "two.sided", exact = FALSE)
```

```
##
##  Wilcoxon rank sum test with continuity correction
##
## data:  useful_rank_1_num by Training.condition
## W = 224197, p-value = 0.381
## alternative hypothesis: true location shift is not equal to 0
```

### 0.4 Equivalence test function

Below creates a function for a two sided equivalence test (TOST) for main effects. It both calculates effect size using cohen's d, using the 90% CI, and then tests each bound for equivalence.

This code is manually created rather than using pre-existing packages (such as TOSTER). This is because it is a mixed-effect model, and the 'lmer' modelling function uses the Satterthwaite approximation to estimate the degrees of freedom for each fixed effect.

Degrees of freedom define the t-distribution used in the test. In mixed-effects models, the presence of clustered or repeated measurements violates the independence assumption, meaning the effective degrees of freedom must be approximated and are typically lower than in simpler models. The Satterthwaite approximation accounts for the uncertainty in estimating random-effects variance components, providing adjusted degrees of freedom appropriate for valid inference.

Interpretation of cohen's d:

- for a main effect, d refers to the effect size of moving from group 0 to 1
- for an interaction, d refers to the standardised difference in slopes between groups (e.g., the slope for when training = 0, and the slope for when training = 1)

```r
tost_from_mixed_model <- function(model, term, sesoi_d = 0.1, alpha = 0.05) {
  # Load required package
  if (!requireNamespace("lmerTest", quietly = TRUE)) {
    stop("Please install the 'lmerTest' package.")
  }

  # Extract total SD across all variance components
  varcomps <- as.data.frame(VarCorr(model))
  sd_total <- sqrt(sum(varcomps$vcov))  # standardise to d-units

  # Extract estimate and SE for the term
  est <- fixef(model)[term]
  se <- summary(model)$coefficients[term, "Std. Error"]
  df <- summary(model)$coefficients[term, "df"]

  # Convert to Cohen's d
  d <- est / sd_total
  se_d <- se / sd_total

  # TOST t-values for one-sided tests at alpha
  t_lower <- (d - (-sesoi_d)) / se_d  # test: d > -sesoi
  t_upper <- (d - sesoi_d) / se_d     # test: d <  sesoi

  # p-values
  p_lower <- pt(t_lower, df, lower.tail = FALSE)
  p_upper <- pt(t_upper, df, lower.tail = TRUE)
```

```
  # 90% CI for d (corresponds to TOST)
  t_crit <- qt(1 - alpha, df)  # captures the 95% percentile for a 90% CI
  ci_lower <- d - t_crit * se_d
  ci_upper <- d + t_crit * se_d

  # Return
  result <- list(
    d = d,
    se_d = se_d,
    df = df,
    t_lower = t_lower,
    t_upper = t_upper,
    ci_90 = c(ci_lower, ci_upper),
    p_lower = p_lower,
    p_upper = p_upper,
    equivalent = (p_lower < alpha & p_upper < alpha)
  )
  class(result) <- "tost_d_result"
  return(result)
}

# Print method
print.tost_d_result <- function(x, ...) {
  cat("TOST for Cohen's d:\n")
  cat(sprintf("  d estimate       = %.3f\n", x$d))
  cat(sprintf("  90%% CI for d     = [%.3f, %.3f]\n", x$ci_90[1], x$ci_90[2]))
  cat(sprintf("  Lower bound test: t(%.1f) = %.2f, p = %.4f\n", x$df, x$t_lower, x$p_lower))
  cat(sprintf("  Upper bound test: t(%.1f) = %.2f, p = %.4f\n", x$df, x$t_upper, x$p_upper))
  cat(sprintf("  Equivalence result: %s\n",
              ifelse(x$equivalent, "EQUIVALENT (within SESOI)", "NOT EQUIVALENT")))
}
```

## 0.5 Hypothesis 1

The underlying structures of the persuasion knowledge scale is checked using a CFA from the package Lavaan.

```
# isolating the PK measures
pk_subset <- mixedeffect_df[, c("PK1_value", "PK2_value", "PK3_value", "PK4_value")]

# creating the items formatted correctly for Lavaan
cfa_pk <- 'pk =~ PK1_value + PK2_value + PK3_value + PK4_value'

# run the CFA
fit_pk <- cfa(cfa_pk, data=pk_subset,
std.lv=T, missing='direct',
estimator='MLR')

# view factor loadings and model fit indices
summary(fit_pk, fit.measures=T)
```

```
## lavaan 0.6-19 ended normally after 21 iterations
##
```

```
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        12
##
##   Number of observations                          5288
##   Number of missing patterns                         1
##
## Model Test User Model:
##                                               Standard      Scaled
##   Test Statistic                              1790.370     825.795
##   Degrees of freedom                                 2           2
##   P-value (Chi-square)                           0.000       0.000
##   Scaling correction factor                                  2.168
##     Yuan-Bentler correction (Mplus variant)
##
## Model Test Baseline Model:
##
##   Test statistic                              9349.940    5135.159
##   Degrees of freedom                                 6           6
##   P-value                                        0.000       0.000
##   Scaling correction factor                                  1.821
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                    0.809       0.839
##   Tucker-Lewis Index (TLI)                       0.426       0.518
##
##   Robust Comparative Fit Index (CFI)                         0.809
##   Robust Tucker-Lewis Index (TLI)                            0.426
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)             -32360.019  -32360.019
##   Scaling correction factor                                  1.504
##       for the MLR correction
##   Loglikelihood unrestricted model (H1)     -31464.834  -31464.834
##   Scaling correction factor                                  1.599
##       for the MLR correction
##
##   Akaike (AIC)                               64744.039   64744.039
##   Bayesian (BIC)                             64822.917   64822.917
##   Sample-size adjusted Bayesian (SABIC)      64784.785   64784.785
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                          0.411       0.279
##   90 Percent confidence interval - lower         0.395       0.268
##   90 Percent confidence interval - upper         0.427       0.290
##   P-value H_0: RMSEA <= 0.050                    0.000       0.000
##   P-value H_0: RMSEA >= 0.080                    1.000       1.000
##
##   Robust RMSEA                                               0.411
##   90 Percent confidence interval - lower                     0.392
##   90 Percent confidence interval - upper                     0.431
```

```
##   P-value H_0: Robust RMSEA <= 0.050                          0.000
##   P-value H_0: Robust RMSEA >= 0.080                          1.000
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                          0.138       0.138
##
## Parameter Estimates:
##
##   Standard errors                             Sandwich
##   Information bread                           Observed
##   Observed information based on                Hessian
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   pk =~
##     PK1_value         1.777    0.021   84.280    0.000
##     PK2_value         0.140    0.013   10.617    0.000
##     PK3_value         1.674    0.022   75.223    0.000
##     PK4_value         0.231    0.015   15.380    0.000
##
## Intercepts:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .PK1_value         3.649    0.026  140.892    0.000
##    .PK2_value         6.226    0.012  499.385    0.000
##    .PK3_value         3.478    0.025  137.980    0.000
##    .PK4_value         6.058    0.014  432.801    0.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .PK1_value         0.389    0.063    6.210    0.000
##    .PK2_value         0.802    0.030   27.185    0.000
##    .PK3_value         0.557    0.059    9.476    0.000
##    .PK4_value         0.983    0.032   30.492    0.000
##     pk                1.000
```

```r
# Comparing the alpha using the psych package

pk_advert <- mixedeffect_df[, c("PK1_value", "PK3_value")]

# extract alpha for all 4 items
alpha4 <- alpha(pk_subset)$total$raw_alpha

# run and extract alpha for 2 items
alpha2 <- alpha(pk_advert)$total$raw_alpha

# present as a comparison table
comparison <- data.frame(
  Items = c("PK1, PK2, PK3, PK4", "PK1, PK3"),
  CronbachAlpha = round(c(alpha4, alpha2), 2)
)

kable(comparison, caption = "Comparison of Cronbach's alpha for 4 items vs 2 items")
```

Table 2: Comparison of Cronbach's alpha for 4 items vs 2 items

| Items | CronbachAlpha |
|---|---|
| PK1, PK2, PK3, PK4 | 0.69 |
| PK1, PK3 | 0.93 |

The decision was made to only analyse the first and third persuasion knowledge item, capturing the distinct subscale of 'advertisement recognition'.

Below runs the model and creates a table using knitr to visualise the effects and model indices.

```
# optimiser is used to help convergence
perc_advert <- lmer(PK_advert ~ version + Training.condition + agree_value + (1 | id) + (1|advert), data

# Extract fixed effects with confidence intervals
fixed_effects <- tidy(perc_advert, effects = "fixed", conf.int = TRUE)

# Extract variance components
random_effects <- as.data.frame(VarCorr(perc_advert))

# Extract  00 values for id and advert
tau00_id <- random_effects[random_effects$grp == "id", "vcov"]
tau00_advert <- random_effects[random_effects$grp == "advert", "vcov"]

# Extract model performance metrics
model_metrics <- model_performance(perc_advert)
icc <- model_metrics$ICC
sigma <- model_metrics$Sigma
r2_marginal <- model_metrics$R2_marginal
r2_conditional <- model_metrics$R2_conditional

# Convert model fit metrics into rows
metric_rows <- tibble(
  term = c(
    "Random Effect id",
    "Random Effect advert",
    "ICC",
    "sigma^2",
    "Marginal R2",
    "Conditional R2"
  ),
  estimate = c(tau00_id, tau00_advert, icc, sigma, r2_marginal, r2_conditional),
  std.error = NA,
  conf.low = NA,
  conf.high = NA,
  p.value = NA
)

# Bind these rows to fixed_effects
fixed_effects <- fixed_effects %>%
  dplyr::select(term, estimate, std.error, conf.low, conf.high, p.value) %>%
  bind_rows(metric_rows)
```

```r
#rounding for better formatting
fixed_effects <- fixed_effects %>%
  mutate(across(
    c(estimate, std.error, conf.low, conf.high, p.value),
    ~ formatC(., format = "f", digits = 3)
  ))

# Rename row terms
fixed_effects <- fixed_effects %>%
  mutate(term = case_when(
    term == "(Intercept)" ~ "Intercept",
    term == "version1" ~ "Digital imprint viewed\n (ref: not viewed)",
    term == "Training.condition1" ~ "Training\n (ref: no training)",
    term == "agree_value" ~ "Agreement",
    TRUE ~ term
  ))

# Replace NA values with blank spaces
fixed_effects <- fixed_effects %>%
  mutate(across(everything(), as.character)) %>%
  mutate(across(where(is.character), ~ gsub("^\\s*NA\\s*$", "", .)))

# necessary for knitting to pdf
fixed_effects <- fixed_effects %>%
  mutate(term = gsub("\\(Intercept\\)", "Intercept", term))

#visualising

fixed_effects %>%
  dplyr::select(
    term, estimate, std.error, conf.low, conf.high, p.value
  ) %>%
  kable(
    caption = "Outcome: persuasion knowledge: advert recognition",
    col.names = c("Term", "Coefficient", "Std. Error", "Lower CI", "Upper CI", "p-value")
  ) %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"), full_width = FALSE)
```

Table 3: Outcome: persuasion knowledge: advert recognition

| Term | Coefficient | Std. Error | Lower CI | Upper CI | p-value |
|---|---|---|---|---|---|
| Intercept | 3.817 | 0.380 | 2.671 | 4.964 | 0.001 |
| version | 0.162 | 0.035 | 0.093 | 0.231 | 0.000 |
| Training.condition | 0.230 | 0.067 | 0.099 | 0.362 | 0.001 |
| Agreement | -0.099 | 0.015 | -0.130 | -0.069 | 0.000 |
| Random Effect id | 1.075 | | | | |
| Random Effect advert | 0.549 | | | | |
| ICC | 0.498 | | | | |
| sigma^2 | 1.280 | | | | |
| Marginal R2 | 0.013 | | | | |
| Conditional R2 | 0.504 | | | | |

Below calculates cohen's d and runs the equivalence test on the main effects.

```
## TOST for Cohen's d:
##   d estimate       = 0.090
##   90% CI for d     = [0.058, 0.122]
##   Lower bound test: t(3961.2) = 9.73, p = 0.0000
##   Upper bound test: t(3961.2) = -0.53, p = 0.2983
##   Equivalence result: NOT EQUIVALENT
```

## 0.6 Hypothesis 2 (exploratory)

Previous studies have predicted a negative effect of a disclosure on perceptions of credibility. We test this in an exploratory way to more clearly situate our findings with other papers. It is noted many studies test an indirect mediation through persuasion knowledge. Mediation analysis relies on independence between data points, and may be not be appropriate with our data. For this reason, we only test a main effect.

```r
# isolate credible scale
cred_subset <- mixedeffect_df[, c("trustworthy_value", "believable_value", "accurate_value", "factual_va

# fit cfa
cfa_credible <- 'credible =~ trustworthy_value + believable_value + accurate_value + factual_value'

fit_credible <- cfa(cfa_credible, data=cred_subset,
std.lv=T, missing='direct',
estimator='MLR')

# view factor loadings and model fit indices
summary(fit_credible, fit.measures=T)
```

```
## lavaan 0.6-19 ended normally after 18 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                        12
##
##   Number of observations                          5288
##   Number of missing patterns                         1
##
## Model Test User Model:
##                                               Standard      Scaled
##   Test Statistic                               255.957     208.837
##   Degrees of freedom                                 2           2
##   P-value (Chi-square)                           0.000       0.000
##   Scaling correction factor                                  1.226
##     Yuan-Bentler correction (Mplus variant)
##
## Model Test Baseline Model:
##
##   Test statistic                             15471.247    8244.147
##   Degrees of freedom                                 6           6
##   P-value                                        0.000       0.000
##   Scaling correction factor                                  1.877
```

```
##
## User Model versus Baseline Model:
##
##   Comparative Fit Index (CFI)                    0.984       0.975
##   Tucker-Lewis Index (TLI)                       0.951       0.925
##
##   Robust Comparative Fit Index (CFI)                         0.984
##   Robust Tucker-Lewis Index (TLI)                            0.951
##
## Loglikelihood and Information Criteria:
##
##   Loglikelihood user model (H0)             -31908.509  -31908.509
##   Scaling correction factor                                  1.308
##       for the MLR correction
##   Loglikelihood unrestricted model (H1)     -31780.531  -31780.531
##   Scaling correction factor                                  1.296
##       for the MLR correction
##
##   Akaike (AIC)                               63841.019   63841.019
##   Bayesian (BIC)                             63919.897   63919.897
##   Sample-size adjusted Bayesian (SABIC)      63881.765   63881.765
##
## Root Mean Square Error of Approximation:
##
##   RMSEA                                          0.155       0.140
##   90 Percent confidence interval - lower         0.139       0.126
##   90 Percent confidence interval - upper         0.171       0.155
##   P-value H_0: RMSEA <= 0.050                    0.000       0.000
##   P-value H_0: RMSEA >= 0.080                    1.000       1.000
##
##   Robust RMSEA                                               0.155
##   90 Percent confidence interval - lower                     0.138
##   90 Percent confidence interval - upper                     0.173
##   P-value H_0: Robust RMSEA <= 0.050                         0.000
##   P-value H_0: Robust RMSEA >= 0.080                         1.000
##
## Standardized Root Mean Square Residual:
##
##   SRMR                                           0.019       0.019
##
## Parameter Estimates:
##
##   Standard errors                             Sandwich
##   Information bread                           Observed
##   Observed information based on                Hessian
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   credible =~
##     trustworthy_vl    1.292    0.015   84.351    0.000
##     believable_val    1.389    0.016   88.349    0.000
##     accurate_value    1.343    0.015   88.762    0.000
##     factual_value     1.234    0.021   57.537    0.000
##
```

```
## Intercepts:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .trustworthy_vl    3.676    0.020  179.817    0.000
##    .believable_val    4.496    0.021  210.929    0.000
##    .accurate_value    4.211    0.020  210.187    0.000
##    .factual_value     3.263    0.025  132.099    0.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .trustworthy_vl    0.541    0.018   30.516    0.000
##    .believable_val    0.474    0.021   22.616    0.000
##    .accurate_value    0.318    0.018   18.066    0.000
##    .factual_value     1.704    0.048   35.442    0.000
##     credible          1.000
```

```r
# Cronbachs alpha
cred_alpha <- alpha(cred_subset)$total$raw_alpha
print(paste("Credibility Cronbach's alpha:", round(cred_alpha, 2)))
```

```
## [1] "Credibility Cronbach's alpha: 0.9"
```

```r
cred <- lmer(credibility ~ Training.condition + version + agree_value + (1|id) + (1|advert), data = mix

# Extract fixed effects with confidence intervals
fixed_effects <- tidy(cred, effects = "fixed", conf.int = TRUE)

# Extract variance components
random_effects <- as.data.frame(VarCorr(cred))

# Extract  00 values for id and advert
tau00_id <- random_effects[random_effects$grp == "id", "vcov"]
tau00_advert <- random_effects[random_effects$grp == "advert", "vcov"]

# Extract model performance metrics
model_metrics <- model_performance(cred)
icc <- model_metrics$ICC
sigma <- model_metrics$Sigma
r2_marginal <- model_metrics$R2_marginal
r2_conditional <- model_metrics$R2_conditional

# Convert model fit metrics into rows
metric_rows <- tibble(
  term = c("Random Effect (id)", "Random Effect (advert)", "ICC", " -squared", "Marginal R-squared", "Co
  estimate = c(tau00_id, tau00_advert, icc, sigma, r2_marginal, r2_conditional),
  std.error = NA,
  conf.low = NA,
  conf.high = NA,
  p.value = NA
)

# Bind these rows to fixed_effects
fixed_effects <- fixed_effects %>%
  dplyr::select(term, estimate, std.error, conf.low, conf.high, p.value) %>%
```

```r
  bind_rows(metric_rows)

#rounding for better formatting
fixed_effects <- fixed_effects %>%
  mutate(across(
    c(estimate, std.error, conf.low, conf.high, p.value),
    ~ formatC(., format = "f", digits = 3)
  ))

# Rename row terms
fixed_effects <- fixed_effects %>%
  mutate(term = case_when(
    term == "(Intercept)" ~ "Intercept",
    term == "version" ~ "Digital imprint viewed\n (ref: not viewed)",
    term == "Training.condition" ~ "Training\n (ref: no training)",
    term == "agree_value" ~ "Agreement with campaign",
    TRUE ~ term
  ))

# Replace NA values with blank spaces
fixed_effects <- fixed_effects %>%
  dplyr::mutate(across(everything(), as.character)) %>%
  dplyr::mutate(across(where(is.character), ~ gsub("^\\s*NA\\s*$", "", .)))

fixed_effects <- fixed_effects %>%
  mutate(term = gsub("\\(Intercept\\)", "Intercept", term))

#visualising

fixed_effects %>%
  dplyr::select(
    term, estimate, std.error, conf.low, conf.high, p.value
  ) %>%
  kable(
    caption = "Outcome: perceived credibility",
    col.names = c("Term", "Coefficient", "Std. Error", "Lower CI", "Upper CI", "p-value")
  ) %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"), full_width = FALSE)
```

Table 4: Outcome: perceived credibility

| Term | Coefficient | Std. Error | Lower CI | Upper CI | p-value |
|---|---|---|---|---|---|
| Intercept | 0.948 | 0.115 | 0.633 | 1.264 | 0.001 |
| Training (ref: no training) | \|0.003 | \|0.032 | \|-0.059 | \|0.065 | \|0.923 |
| Digital imprint viewed (ref: not viewed) | \|0.069 | \|0.021 | \|0.028 | \|0.109 | \|0.001 |
| Agreement with campaign | 0.645 | 0.009 | 0.628 | 0.662 | 0.000 |
| Random Effect (id) | 0.194 | | | | |
| Random Effect (advert) | 0.045 | | | | |
| ICC | 0.299 | | | | |
| -squared | 0.747 | | | | |
| Marginal R-squared | 0.546 | | | | |

| | |
|---|---|
| Conditional R-squared | 0.682 |

We find an unexpected positive effect of the disclosure on perceptions of credibility. Below tests the effect for practical significance as well as the robustness of the p-value using the false discovery rate method.

```
## TOST for Cohen's d:
##   d estimate       = 0.077
##   90% CI for d     = [0.039, 0.115]
##   Lower bound test: t(3927.6) = 7.69, p = 0.0000
##   Upper bound test: t(3927.6) = -1.00, p = 0.1598
##   Equivalence result: NOT EQUIVALENT
```

Table 5: Original and Adjusted p-Values for Fixed Effects

| Fixed Effect | Original P-Value | Adjusted P-Value |
|---|---|---|
| Intercept | 0.001 | 0.001 |
| Training (ref: no training) | \|0.923 | \|0.923 |
| Digital imprint viewed (ref: not viewed) | \|0.001 | \|0.001 |
| agree_value | 0.000 | 0.000 |

## 0.7 Hypothesis 3

```r
cred_int <- lmer(credibility ~ Training.condition*version + agree_value + (1|id) + (1|advert), data = mi

# Extract fixed effects with confidence intervals
fixed_effects <- tidy(cred_int, effects = "fixed", conf.int = TRUE)

# Extract variance components
random_effects <- as.data.frame(VarCorr(cred_int))

# Extract  00 values for id and advert
tau00_id <- random_effects[random_effects$grp == "id", "vcov"]
tau00_advert <- random_effects[random_effects$grp == "advert", "vcov"]

# Extract model performance metrics
model_metrics <- model_performance(cred_int)
icc <- model_metrics$ICC
sigma <- model_metrics$Sigma
r2_marginal <- model_metrics$R2_marginal
r2_conditional <- model_metrics$R2_conditional

# Bind these rows to fixed_effects
fixed_effects <- fixed_effects %>%
  dplyr::select(term, estimate, std.error, conf.low, conf.high, p.value) %>%
  bind_rows(metric_rows)

#rounding for better formatting
fixed_effects <- fixed_effects %>%
  mutate(across(
```

```r
    c(estimate, std.error, conf.low, conf.high, p.value),
    ~ formatC(., format = "f", digits = 3)
  ))

# Rename row terms
fixed_effects <- fixed_effects %>%
  mutate(term = case_when(
    term == "(Intercept)" ~ "Intercept",
    term == "version" ~ "Digital imprint viewed\n (ref: not viewed)",
    term == "Training.condition" ~ "Training\n (ref: no training)",
    term == "Training.condition:version" ~ "Training*Version",
    term == "agree_value" ~ "Agreement with campaign",
    TRUE ~ term
  ))

# Replace NA values with blank spaces
fixed_effects <- fixed_effects %>%
  dplyr::mutate(across(everything(), as.character)) %>%
  dplyr::mutate(across(where(is.character), ~ gsub("^\\s*NA\\s*$", "", .)))
fixed_effects <- fixed_effects %>%
  mutate(term = gsub("\\(Intercept\\)", "Intercept", term))

#visualising

fixed_effects %>%
  dplyr::select(
    term, estimate, std.error, conf.low, conf.high, p.value
  ) %>%
  kable(
    caption = "Outcome: perceived credibility",
    col.names = c("Term", "Coefficient", "Std. Error", "Lower CI", "Upper CI", "p-value")
  ) %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"), full_width = FALSE)
```

Table 6: Outcome: perceived credibility

| Term | Coefficient | Std. Error | Lower CI | Upper CI | p-value |
|---|---|---|---|---|---|
| Intercept | 0.958 | 0.116 | 0.643 | 1.273 | 0.001 |
| Training (ref: no training) | \|-0.016 | \|0.038 | \|-0.091 | \|0.058 | \|0.665 |
| Digital imprint viewed (ref: not viewed) | \|0.049 | \|0.029 | \|-0.008 | \|0.106 | \|0.091 |
| Agreement with campaign | 0.645 | 0.009 | 0.628 | 0.662 | 0.000 |
| Training*Version | 0.039 | 0.041 | -0.042 | 0.119 | 0.344 |
| Random Effect (id) | 0.194 | | | | |
| Random Effect (advert) | 0.045 | | | | |
| ICC | 0.299 | | | | |
| -squared | 0.747 | | | | |
| Marginal R-squared | 0.546 | | | | |
| Conditional R-squared | 0.682 | | | | |

```
## TOST for Cohen's d:
##   d estimate       = 0.044
```

```
##   90% CI for d     = [-0.032, 0.119]
##   Lower bound test: t(3925.9) = 3.12, p = 0.0009
##   Upper bound test: t(3925.9) = -1.22, p = 0.1103
##   Equivalence result: NOT EQUIVALENT
```

The code below checks model fit for the simpler (no interaction) and interaction model. It uses the AIC and Likelihood Ratio Test (LRT) to compare models. A lower AIC value is better. It can be seen the interaction effect worsens model parsimony (AIC) and does not improve model fit (LRT).

Table 7: AIC: perceived credibility

| | Degrees of Freedom | AIC |
|---|---|---|
| version + training + agree | 7 | 13120.21 |
| version*training + agree | 8 | 13125.86 |

```
## Contrasts set to contr.sum for the following variables: advert

## Numerical variables NOT centered on 0: Training.condition, version, agree_value
## If in interactions, interpretation of lower order (e.g., main) effects difficult.

## REML argument to lmer() set to FALSE for method = 'PB' or 'LRT'

## Mixed Model Anova Table (Type 3 tests, LRT-method)
##
## Model: credibility ~ Training.condition * version + agree_value + (1 |
## Model:     id) + (1 | advert)
## Data: mixedeffect_df
## Df full model: 8
##                       Effect df       Chisq p.value
## 1         Training.condition  1        0.19    .665
## 2                    version  1      2.87 +    .090
## 3                agree_value  1 3772.95 ***   <.001
## 4 Training.condition:version  1        0.90    .344
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '+' 0.1 ' ' 1
```

## 0.8 Hypothesis 4

Effect of version on perceived self-informedness.

```
#perceived self-informedness
#with only 3 items, this is a just identified model so model fit indices cannot be checked. Instead, fa

in_subset <- mixedeffect_df[, c("informed2_value", "informed3_value", "informed4_value")]

cfa_informed <- 'informed =~ informed2_value + informed3_value + informed4_value'

# fit CFA
fit_informed <- cfa(cfa_informed, data=in_subset,
std.lv=T, missing='direct',
estimator='MLR')
```

```r
# view factor loadings
summary(fit_informed)
```

```
## lavaan 0.6-19 ended normally after 15 iterations
##
##   Estimator                                         ML
##   Optimization method                           NLMINB
##   Number of model parameters                         9
##
##   Number of observations                          5288
##   Number of missing patterns                         1
##
## Model Test User Model:
##                                               Standard      Scaled
##   Test Statistic                                 0.000       0.000
##   Degrees of freedom                                 0           0
##
## Parameter Estimates:
##
##   Standard errors                             Sandwich
##   Information bread                           Observed
##   Observed information based on                Hessian
##
## Latent Variables:
##                    Estimate  Std.Err  z-value  P(>|z|)
##   informed =~
##     informed2_valu    1.596    0.016   98.777    0.000
##     informed3_valu    1.432    0.021   69.153    0.000
##     informed4_valu    1.399    0.018   78.168    0.000
##
## Intercepts:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .informed2_valu    4.203    0.024  174.168    0.000
##    .informed3_valu    3.992    0.025  160.015    0.000
##    .informed4_valu    4.422    0.023  194.801    0.000
##
## Variances:
##                    Estimate  Std.Err  z-value  P(>|z|)
##    .informed2_valu    0.531    0.033   16.110    0.000
##    .informed3_valu    1.242    0.052   23.896    0.000
##    .informed4_valu    0.767    0.033   23.133    0.000
##     informed          1.000
```

```r
# Cronbachs alpha
inform_alpha <- alpha(in_subset)$total$raw_alpha
print(paste("Subjective informedness Cronbach's alpha:", round(inform_alpha, 2)))
```

```
## [1] "Subjective informedness Cronbach's alpha: 0.88"
```

```r
informed_model <- lmer(informed ~ Training.condition + version + agree_value + (1|id) + (1|advert), data

# Extract fixed effects with confidence intervals
```

```r
fixed_effects <- tidy(informed_model, effects = "fixed", conf.int = TRUE)

# Extract variance components
random_effects <- as.data.frame(VarCorr(informed_model))

# Extract  00 values for id and advert
tau00_id <- random_effects[random_effects$grp == "id", "vcov"]
tau00_advert <- random_effects[random_effects$grp == "advert", "vcov"]

# Extract model performance metrics
model_metrics <- model_performance(informed_model)
icc <- model_metrics$ICC
sigma <- model_metrics$Sigma
r2_marginal <- model_metrics$R2_marginal
r2_conditional <- model_metrics$R2_conditional

# Bind these rows to fixed_effects
fixed_effects <- fixed_effects %>%
  dplyr::select(term, estimate, std.error, conf.low, conf.high, p.value) %>%
  bind_rows(metric_rows)

#rounding for better formatting
fixed_effects <- fixed_effects %>%
  mutate(across(
    c(estimate, std.error, conf.low, conf.high, p.value),
    ~ formatC(., format = "f", digits = 3)
  ))

# Rename row terms
fixed_effects <- fixed_effects %>%
  mutate(term = case_when(
    term == "(Intercept)" ~ "Intercept",
    term == "version" ~ "Digital imprint viewed\n (ref: not viewed)",
    term == "Training.condition" ~ "Training\n (ref: no training)",
    term == "agree_value" ~ "Agreement with campaign",
    TRUE ~ term
  ))

# Replace NA values with blank spaces
fixed_effects <- fixed_effects %>%
  dplyr::mutate(across(everything(), as.character)) %>%
  dplyr::mutate(across(where(is.character), ~ gsub("^\\s*NA\\s*$", "", .)))

fixed_effects <- fixed_effects %>%
  mutate(term = gsub("\\(Intercept\\)", "Intercept", term))

#visualising

fixed_effects %>%
  dplyr::select(
    term, estimate, std.error, conf.low, conf.high, p.value
  ) %>%
  kable(
```

```
    caption = "Outcome: perceived self-informedness",
    col.names = c("Term", "Coefficient", "Std. Error", "Lower CI", "Upper CI", "p-value")
  ) %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"), full_width = FALSE)
```

Table 8: Outcome: perceived self-informedness

| Term | Coefficient | Std. Error | Lower CI | Upper CI | p-value |
|---|---|---|---|---|---|
| Intercept | 3.447 | 0.213 | 2.856 | 4.037 | 0.000 |
| Training (ref: no training) | \|0.044 | \|0.056 | \|-0.066 | \|0.154 | \|0.429 |
| Digital imprint viewed (ref: not viewed) | \|0.323 | \|0.036 | \|0.253 | \|0.393 | \|0.000 |
| Agreement with campaign | 0.127 | 0.015 | 0.098 | 0.156 | 0.000 |
| Random Effect (id) | 0.194 | | | | |
| Random Effect (advert) | 0.045 | | | | |
| ICC | 0.299 | | | | |
| -squared | 0.747 | | | | |
| Marginal R-squared | 0.546 | | | | |
| Conditional R-squared | 0.682 | | | | |

## 0.9 Hypothesis 5

Effect of version x training on perceived self-informedness.

```
informed_int <- lmer(informed ~ Training.condition*version + agree_value + (1|id) + (1|advert), data = r

# Extract fixed effects with confidence intervals
fixed_effects <- tidy(informed_int, effects = "fixed", conf.int = TRUE)

# Extract variance components
random_effects <- as.data.frame(VarCorr(informed_int))

# Extract  00 values for id and advert
tau00_id <- random_effects[random_effects$grp == "id", "vcov"]
tau00_advert <- random_effects[random_effects$grp == "advert", "vcov"]

# Extract model performance metrics
model_metrics <- model_performance(informed_int)
icc <- model_metrics$ICC
sigma <- model_metrics$Sigma
r2_marginal <- model_metrics$R2_marginal
r2_conditional <- model_metrics$R2_conditional

# Bind these rows to fixed_effects
fixed_effects <- fixed_effects %>%
  dplyr::select(term, estimate, std.error, conf.low, conf.high, p.value) %>%
  bind_rows(metric_rows)

#rounding for better formatting
fixed_effects <- fixed_effects %>%
  mutate(across(
    c(estimate, std.error, conf.low, conf.high, p.value),
```

```r
    ~ formatC(., format = "f", digits = 3)
  ))

# Rename row terms
fixed_effects <- fixed_effects %>%
  mutate(term = case_when(
    term == "(Intercept)" ~ "Intercept",
    term == "version" ~ "Digital imprint viewed\n (ref: not viewed)",
    term == "Training.condition" ~ "Training\n (ref: no training)",
    term == "agree_value" ~ "Agreement",
    term == "Training.condition:version" ~ "Training*Version",
    TRUE ~ term
  ))

# Replace NA values with blank spaces
fixed_effects <- fixed_effects %>%
  mutate(across(everything(), as.character)) %>%
  mutate(across(where(is.character), ~ gsub("^\\s*NA\\s*$", "", .)))

fixed_effects <- fixed_effects %>%
  mutate(term = gsub("\\(Intercept\\)", "Intercept", term))

#visualising

fixed_effects %>%
  dplyr::select(
    term, estimate, std.error, conf.low, conf.high, p.value
  ) %>%
  kable(
    caption = "Outcome: perceived self informedness",
    col.names = c("Term", "Coefficient", "Std. Error", "Lower CI", "Upper CI", "p-value")
  ) %>%
  kable_styling(bootstrap_options = c("striped", "hover", "condensed"), full_width = FALSE)
```

Table 9: Outcome: perceived self informedness

| Term | Coefficient | Std. Error | Lower CI | Upper CI | p-value |
|---|---|---|---|---|---|
| Intercept | 3.501 | 0.214 | 2.913 | 4.090 | 0.000 |
| Training (ref: no training) | \|-0.067 | \|0.066 | \|-0.197 | \|0.064 | \|0.316 |
| Digital imprint viewed (ref: not viewed) | \|0.212 | \|0.051 | \|0.113 | \|0.311 | \|0.000 |
| Agreement | 0.127 | 0.015 | 0.098 | 0.156 | 0.000 |
| Training*Version | 0.222 | 0.071 | 0.082 | 0.362 | 0.002 |
| Random Effect (id) | 0.194 | | | | |
| Random Effect (advert) | 0.045 | | | | |
| ICC | 0.299 | | | | |
| -squared | 0.747 | | | | |
| Marginal R-squared | 0.546 | | | | |
| Conditional R-squared | 0.682 | | | | |

The code below checks model fit for the simpler (no interaction) and interaction model. It uses the AIC and Likelihood Ratio Test to compare both models. Lower AIC values indicate a better fitting model.

Table 10: AIC: perceived self-informedness

| | Degrees of Freedom | AIC |
|---|---|---|
| version + training + agree | 7 | 19016.13 |
| version*training + agree | 8 | 19011.94 |

```
## Contrasts set to contr.sum for the following variables: advert
```

```
## Numerical variables NOT centered on 0: Training.condition, version, agree_value
## If in interactions, interpretation of lower order (e.g., main) effects difficult.
```

```
## REML argument to lmer() set to FALSE for method = 'PB' or 'LRT'
```

```
## Mixed Model Anova Table (Type 3 tests, LRT-method)
##
## Model: informed ~ Training.condition * version + agree_value + (1 |
## Model:     id) + (1 | advert)
## Data: mixedeffect_df
## Df full model: 8
##                       Effect df     Chisq p.value
## 1         Training.condition  1      1.01    .316
## 2                    version  1 17.51 ***   <.001
## 3                agree_value  1 71.98 ***   <.001
## 4 Training.condition:version  1   9.63 **    .002
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '+' 0.1 ' ' 1
```

Equivalence for the main effect of version in model 5 (the more valid estimate of this effect):

```
## TOST for Cohen's d:
##   d estimate       = 0.135
##   90% CI for d     = [0.082, 0.188]
##   Lower bound test: t(3959.4) = 7.29, p = 0.0000
##   Upper bound test: t(3959.4) = 1.09, p = 0.8621
##   Equivalence result: NOT EQUIVALENT
```

Equivalence for the interaction effect in model 5:

```
## TOST for Cohen's d:
##   d estimate       = 0.141
##   90% CI for d     = [0.066, 0.216]
##   Lower bound test: t(3959.3) = 5.30, p = 0.0000
##   Upper bound test: t(3959.3) = 0.91, p = 0.8186
##   Equivalence result: NOT EQUIVALENT
```

## 0.10 Hypothesis 6

Effect of the training on confidence in regulatory effectiveness.

```r
regulation_model <- lm(election_reg ~ Training.condition, data = betweenmeasure_df)

summary(regulation_model)
```

```
##
## Call:
## lm(formula = election_reg ~ Training.condition, data = betweenmeasure_df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.0091 -1.0091 -0.0091  0.9909  4.1041
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)         3.00910    0.05571  54.014   <2e-16 ***
## Training.condition -0.11318    0.07867  -1.439     0.15
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.43 on 1320 degrees of freedom
## Multiple R-squared:  0.001566,   Adjusted R-squared:  0.0008092
## F-statistic:  2.07 on 1 and 1320 DF,  p-value: 0.1505

## Cohen's d: d = -0.079 , 90% CI: [ -0.17 ,  0.011 ]

## Lower bound test:  t = 0.38 , p = 0.352

## Upper bound test:  t = -3.26 , p = 6e-04
```

### 0.11 Analysis Specific References


Bakdash JZ and Marusich LR (2017) Repeated measures correlation. Frontiers in Psychology 8: 1-13.

Brown VA (2021) An introduction to linear mixed-effects modeling in R. Advances in Methods and Practices in Psychological Science 4(1): 2515245920960351.

Isager (2019) "Mixed model equivalence test using R and PANGEA". Link: https://pedermisager.org/blog/mixed_model_equivalence/

Lakens D (2024) When and how to deviate from a preregistration. Collabra: Psychology 10(1): 117094.

Lakens, D., Scheel, A.M. and Isager, P.M., 2018. Equivalence testing for psychological research: A tutorial. Advances in methods and practices in psychological science, 1(2), pp.259-269.

Matuschek H, Kliegl R, Vasishth S, Baayen H and Bates D (2017) Balancing Type I error and power in linear mixed models. Journal of Memory and Language 94: 305-315.

Singmann H and Kellen D (2019) An Introduction to Mixed Models for Experimental Psychology. In: Spieler DH and Schumacher E (eds) New Methods in Cognitive Psychology. New York and London: Routledge, pp. 4-31.