Randomized Attack

```
library(conflicted)

library(kableExtra)
```

```
## Warning in !is.null(rmarkdown::metadata$output) && rmarkdown::metadata$output
## %in% : 'length(x) = 2 > 1' in coercion to 'logical(1)'
```

```
library(knitr)
library(broom.helpers)
library(broom)
library(dtplyr)
library(furrr)
```

```
## Loading required package: future
```

```
library(arrow)
library(glue)
library(fs)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
```

```
## v ggplot2 3.4.0      v purrr   1.0.1
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.5.0
## v readr   2.1.3      v forcats 0.5.2
```

```
conflict_prefer("filter", "dplyr")
```

```
## [conflicted] Will prefer dplyr::filter over any other package
```

```
source("./analysis/utils.R", local = knit_global())
set_theme()
```

```
write_bib(.packages(), glue("./analysis/packages.bib"))
```

```
## Warning in utils::citation(..., lib.loc = lib.loc): no date field in DESCRIPTION
## file of package 'kableExtra'
```

```
sessionInfo()
```

```
## R version 4.2.2 (2022-10-31)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Big Sur ... 10.16
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.2/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.2/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
```

```
##
## attached base packages:
## [1] stats     graphics  grDevices datasets  utils     methods   base
##
## other attached packages:
##  [1] forcats_0.5.2         stringr_1.5.0         dplyr_1.0.10
##  [4] purrr_1.0.1           readr_2.1.3           tidyr_1.2.1
##  [7] tibble_3.1.8          ggplot2_3.4.0         tidyverse_1.3.2
## [10] fs_1.5.2              glue_1.6.2            arrow_10.0.1
## [13] furrr_0.3.1           future_1.30.0         dtplyr_1.2.2
## [16] broom_1.0.2           broom.helpers_1.11.0  knitr_1.41
## [19] kableExtra_1.3.4.9000 conflicted_1.1.0
##
## loaded via a namespace (and not attached):
##  [1] httr_1.4.4          bit64_4.0.5         jsonlite_1.8.4
##  [4] viridisLite_0.4.1   here_1.0.1          modelr_0.1.10
##  [7] assertthat_0.2.1    renv_0.16.0         googlesheets4_1.0.1
## [10] cellranger_1.1.0    yaml_2.3.6          globals_0.16.2
## [13] pillar_1.8.1        backports_1.4.1     digest_0.6.31
## [16] rvest_1.0.3         colorspace_2.0-3    htmltools_0.5.4
## [19] pkgconfig_2.0.3     listenv_0.9.0       haven_2.5.1
## [22] scales_1.2.1        webshot_0.5.4       svglite_2.1.1
## [25] tzdb_0.3.0          timechange_0.2.0    googledrive_2.0.0
## [28] generics_0.1.3      ellipsis_0.3.2      withr_2.5.0
## [31] cachem_1.0.6        cli_3.6.0           crayon_1.5.2
## [34] readxl_1.4.1        magrittr_2.0.3      memoise_2.0.1
## [37] evaluate_0.19       fansi_1.0.3         parallelly_1.34.0
## [40] xml2_1.3.3          tools_4.2.2         data.table_1.14.6
## [43] hms_1.1.2           gargle_1.2.1        lifecycle_1.0.3
## [46] reprex_2.0.2        munsell_0.5.0       compiler_4.2.2
## [49] systemfonts_1.0.4   rlang_1.0.6         grid_4.2.2
## [52] rstudioapi_0.14     rmarkdown_2.19      gtable_0.3.1
## [55] codetools_0.2-18    DBI_1.1.3           R6_2.5.1
## [58] lubridate_1.9.0     fastmap_1.1.0       bit_4.0.5
## [61] utf8_1.2.2          rprojroot_2.0.3     stringi_1.7.12
## [64] parallel_4.2.2      vctrs_0.5.1         dbplyr_2.2.1
## [67] tidyselect_1.2.0    xfun_0.36
```

```
data_dir <- path(glue("./data/{params$simulation}/results"))

success_fnames <-
  dir_ls(data_dir, glob = glue("*{params$simulation}*_trend.csv"))

# every fname is a simulation
success_raw_data <- get_data(success_fnames, read_csv) |>
  glimpse()
```

```
## Rows: 3,000
## Columns: 12
## $ fname                  <chr> "./data/random/results/random_repeat_10_mislab~
## $ num_iteration          <dbl> 10, 50, 100, 200, 10, 50, 100, 200, 10, 50, 10~
## $ attack_count           <dbl> 100, 100, 100, 100, 100, 100, 100, 100, 100, 1~
## $ success_count          <dbl> 1, 3, 3, 3, 1, 1, 3, 3, 0, 3, 3, 3, 7, 18, 20,~
## $ vanish_count           <dbl> 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 2, 4, 6, 5~
## $ mislabel_count         <dbl> 0, 3, 3, 3, 0, 1, 3, 3, 0, 3, 3, 3, 5, 14, 14,~
```

```
## $ mislabel_intended_count <dbl> 0, 3, 3, 3, 0, 1, 2, 2, 0, 3, 3, 3, 5, 14, 14,~
## $ sample_count            <dbl> 125, 125, 125, 125, 126, 126, 126, 126, 130, 1~
## $ model_name              <ord> Cascade R-CNN, Cascade R-CNN, Cascade R-CNN, C~
## $ loss_target             <ord> Mislabeling, Mislabeling, Mislabeling, Mislabe~
## $ attack_bbox             <chr> "ground_truth", "ground_truth", "ground_truth"~
## $ perturb_fun             <chr> "perturb_inside", "perturb_inside", "perturb_i~
```

```
# expand success per simulation into 1 and 0s per row
success_expanded_data <- success_raw_data |>
  rowwise() |>
  mutate(success = list(rep(0:1, times = c(attack_count - success_count, success_count)))) |>
  unnest_longer(success) |>
  glimpse()
```

```
## Rows: 300,000
## Columns: 13
## $ fname                   <chr> "./data/random/results/random_repeat_10_mislab~
## $ num_iteration           <dbl> 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10, 10~
## $ attack_count            <dbl> 100, 100, 100, 100, 100, 100, 100, 100, 100, 1~
## $ success_count           <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
## $ vanish_count            <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
## $ mislabel_count          <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ mislabel_intended_count <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ sample_count            <dbl> 125, 125, 125, 125, 125, 125, 125, 125, 125, 1~
## $ model_name              <ord> Cascade R-CNN, Cascade R-CNN, Cascade R-CNN, C~
## $ loss_target             <ord> Mislabeling, Mislabeling, Mislabeling, Mislabe~
## $ attack_bbox             <chr> "ground_truth", "ground_truth", "ground_truth"~
## $ perturb_fun             <chr> "perturb_inside", "perturb_inside", "perturb_i~
## $ success                 <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
```

```
# check whether attack count equals experiment settings
stopifnot(all(success_raw_data$attack_count == 100))

reps <- success_raw_data |>
  count(model_name, loss_target, num_iteration) |>
  glimpse()
```

```
## Rows: 60
## Columns: 4
## $ model_name    <ord> YOLOv3, YOLOv3, YOLOv3, YOLOv3, YOLOv3, YOLOv3, YOLOv3, ~
## $ loss_target   <ord> Vanishing, Vanishing, Vanishing, Vanishing, Mislabeling,~
## $ num_iteration <dbl> 10, 50, 100, 200, 10, 50, 100, 200, 10, 50, 100, 200, 10~
## $ n             <int> 50, 50, 50, 50, 50, 50, 50, 50, 50, 50, 50, 50, 50, 50, ~
```

```
stopifnot(unique(reps$n) == 50)
```

```
itr_lab <- "Attack Iterations"

cap <- glue("{bold_tex('Intent obfuscating attack is feasible for all models and attacks')} We conduct a

cap
```

```
## \textbf{Intent obfuscating attack is feasible for all models and attacks:}  We conduct a randomized c
```

```
# use log(num_iteration)
g <- success_expanded_data |>
  ggplot(aes(num_iteration, success, color = loss_target, linetype = loss_target)) +
```

```
  # use stat_summary rather than stat_summary_bin
  # since num_iteration is set experimentally
  # mean_cl_boot gives 95% bootstrapped CI at 1000 samples
  # https://rdrr.io/cran/Hmisc/man/smean.sd.html
  stat_summary(fun.data = "mean_cl_boot") +
  binomial_smooth(formula = y ~ log(x)) +
  facet_grid(cols = vars(model_name))

g +
  labs(x = itr_lab, y = "p(Success)", color = "Attack", linetype = "Attack") +
  scale_x_continuous(breaks = unique(success_raw_data$num_iteration))
```

```
success_breakdown_data <- success_raw_data |>
  rowwise() |>
  mutate(
    vanish = list(rep(0:1, times = c(success_count - vanish_count, vanish_count))),
    mislabel = list(rep(0:1, times = c(success_count - mislabel_count, mislabel_count)))
  ) |>
  unnest_longer(c(vanish, mislabel)) |>
  pivot_longer(c(vanish, mislabel)) |>
  mutate(name = factor(recode(name, vanish = "Vanished", mislabel = "Mislabeled"), ordered = TRUE)) |>
  glimpse()
```

```
## Rows: 60,016
## Columns: 14
## $ fname                   <chr> "./data/random/results/random_repeat_10_mislab~
## $ num_iteration           <dbl> 10, 10, 50, 50, 50, 50, 50, 50, 100, 100, 100,~
## $ attack_count            <dbl> 100, 100, 100, 100, 100, 100, 100, 100, 100, 1~
## $ success_count           <dbl> 1, 1, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3~
## $ vanish_count            <dbl> 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ mislabel_count          <dbl> 0, 0, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3~
## $ mislabel_intended_count <dbl> 0, 0, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3~
## $ sample_count            <dbl> 125, 125, 125, 125, 125, 125, 125, 125, 125, 1~
## $ model_name              <ord> Cascade R-CNN, Cascade R-CNN, Cascade R-CNN, C~
## $ loss_target             <ord> Mislabeling, Mislabeling, Mislabeling, Mislabe~
## $ attack_bbox             <chr> "ground_truth", "ground_truth", "ground_truth"~
## $ perturb_fun             <chr> "perturb_inside", "perturb_inside", "perturb_i~
## $ name                    <ord> Vanished, Mislabeled, Vanished, Mislabeled, Va~
## $ value                   <int> 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1, 0, 1~
```

```
cap <- glue("{bold_tex('Vanishing and mislabeling attacks mostly cause target objects to vanish and get

cap
```

```
## \textbf{Vanishing and mislabeling attacks mostly cause target objects to vanish and get mislabeled:}
```

```
legend_lab <- "Success Rationale"

g <- success_breakdown_data |>
  ggplot(aes(num_iteration, value, color = name, linetype = name)) +
  # use stat_summary rather than stat_summary_bin
  # since num_iteration is set experimentally
  # mean_cl_boot gives 95% bootstrapped CI at 1000 samples
  # https://rdrr.io/cran/Hmisc/man/smean.sd.html
  stat_summary(fun.data = "mean_cl_boot") +
```

```
  binomial_smooth(formula = y ~ log(x)) +
  facet_grid(cols = vars(model_name), rows = vars(loss_target))

g +
  labs(x = itr_lab, y = "p(Success Rationale)", color = legend_lab, linetype = legend_lab) +
  scale_x_continuous(breaks = unique(success_raw_data$num_iteration)) +
  coord_cartesian(ylim = c(0, 1))
```

```
mislabel_intended_data <- success_raw_data |>
  filter(loss_target == "Mislabeling") |>
  rowwise() |>
  mutate(
    mislabel_intended = list(rep(0:1, times = c(mislabel_count - mislabel_intended_count, mislabel_inter
  ) |>
  unnest_longer(mislabel_intended) |>
  glimpse()
```

```
## Rows: 5,672
## Columns: 13
## $ fname                   <chr> "./data/random/results/random_repeat_10_mislab~
## $ num_iteration           <dbl> 50, 50, 50, 100, 100, 100, 200, 200, 200, 50, ~
## $ attack_count            <dbl> 100, 100, 100, 100, 100, 100, 100, 100, 100, 1~
## $ success_count           <dbl> 3, 3, 3, 3, 3, 3, 3, 3, 3, 1, 3, 3, 3, 3, 3, 3~
## $ vanish_count            <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ mislabel_count          <dbl> 3, 3, 3, 3, 3, 3, 3, 3, 3, 1, 3, 3, 3, 3, 3, 3~
## $ mislabel_intended_count <dbl> 3, 3, 3, 3, 3, 3, 3, 3, 3, 1, 2, 2, 2, 2, 2, 2~
## $ sample_count            <dbl> 125, 125, 125, 125, 125, 125, 125, 125, 125, 1~
## $ model_name              <ord> Cascade R-CNN, Cascade R-CNN, Cascade R-CNN, C~
## $ loss_target             <ord> Mislabeling, Mislabeling, Mislabeling, Mislabe~
## $ attack_bbox             <chr> "ground_truth", "ground_truth", "ground_truth"~
## $ perturb_fun             <chr> "perturb_inside", "perturb_inside", "perturb_i~
## $ mislabel_intended       <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1~
```

```
cap <- glue("{bold_tex('Mislabeling attacks usually mislabel the target objects to the intended class')}

cap
```

```
## \textbf{Mislabeling attacks usually mislabel the target objects to the intended class:}  The binned s
```

```
g <- mislabel_intended_data |>
  ggplot(aes(num_iteration, mislabel_intended)) +
  # use stat_summary rather than stat_summary_bin
  # since num_iteration is set experimentally
  # mean_cl_boot gives 95% bootstrapped CI at 1000 samples
  # https://rdrr.io/cran/Hmisc/man/smean.sd.html
  stat_summary(fun.data = "mean_cl_boot") +
  binomial_smooth(formula = y ~ log(x)) +
  facet_grid(cols = vars(model_name), rows = vars(loss_target))

g +
  labs(x = itr_lab, y = "p(mislabeled to intended class within\nsuccess cases in mislabeling attack)") +
  scale_x_continuous(breaks = unique(success_raw_data$num_iteration)) +
  coord_cartesian(ylim = c(0, 1))
```

```
## Warning: glm.fit: algorithm did not converge
```

```
mislabel_intended_data |> group_by(model_name, num_iteration) |> summarize(mean(mislabel_intended))
```

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

```
## # A tibble: 20 x 3
## # Groups:   model_name [5]
##    model_name    num_iteration `mean(mislabel_intended)`
##    <ord>                 <dbl>                     <dbl>
##  1 YOLOv3                   10                     0.969
##  2 YOLOv3                   50                     0.957
##  3 YOLOv3                  100                     0.975
##  4 YOLOv3                  200                     0.970
##  5 SSD                      10                     0.977
##  6 SSD                      50                     0.999
##  7 SSD                     100                     0.999
##  8 SSD                     200                     0.999
##  9 RetinaNet                10                     0.893
## 10 RetinaNet                50                     0.951
## 11 RetinaNet               100                     0.968
## 12 RetinaNet               200                     0.982
## 13 Faster R-CNN             10                     0.92
## 14 Faster R-CNN             50                     0.971
## 15 Faster R-CNN            100                     0.977
## 16 Faster R-CNN            200                     0.980
## 17 Cascade R-CNN            10                     1
## 18 Cascade R-CNN            50                     1
## 19 Cascade R-CNN           100                     1
## 20 Cascade R-CNN           200                     1
```

```
# compare models against YOLO
# grouped by attack
data <- success_expanded_data |>
  # restrict to max iteration
  filter(num_iteration == max(num_iteration)) |>
  # avoid ordered regression
  mutate(
    model_name = factor(model_name, ordered = FALSE),
    loss_target = factor(loss_target, ordered = FALSE)
  ) |>
  glimpse()
```

```
## Rows: 75,000
## Columns: 13
## $ fname                  <chr> "./data/random/results/random_repeat_10_mislab~
## $ num_iteration          <dbl> 200, 200, 200, 200, 200, 200, 200, 200, 200, 2~
## $ attack_count           <dbl> 100, 100, 100, 100, 100, 100, 100, 100, 100, 1~
## $ success_count          <dbl> 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3~
## $ vanish_count           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ mislabel_count         <dbl> 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3~
## $ mislabel_intended_count <dbl> 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3~
## $ sample_count           <dbl> 125, 125, 125, 125, 125, 125, 125, 125, 125, 1~
## $ model_name             <fct> Cascade R-CNN, Cascade R-CNN, Cascade R-CNN, C~
## $ loss_target            <fct> Mislabeling, Mislabeling, Mislabeling, Mislabe~
## $ attack_bbox            <chr> "ground_truth", "ground_truth", "ground_truth"~
```

```
## $ perturb_fun             <chr> "perturb_inside", "perturb_inside", "perturb_i~
## $ success                 <int> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
```

```
model <- partial(glm_model, predictor = "model_name")

reg_est <- get_tidied_reg(
  model, data, loss_target
)
```

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

```
ext_sig(reg_est)
```

```
## Total 15 predictors:
## 10 (67%) significant;
## 10 (67%) both
```

```
## # A tibble: 10 x 8
## # Groups:   loss_target [3]
##    loss_target term              estim~1 std.e~2 stati~3 p.value conf.~4 conf.~5
##    <fct>       <chr>               <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1 Vanishing   model_nameRetina~  -1.49    0.06   -24.9    0      -1.61   -1.38
##  2 Vanishing   model_nameFaster~  -2.16    0.075  -28.7    0      -2.31   -2.02
##  3 Vanishing   model_nameCascad~  -1.79    0.066  -27.1    0      -1.92   -1.66
##  4 Mislabeling model_nameSSD      -0.283   0.046   -6.18   0      -0.372  -0.193
##  5 Mislabeling model_nameRetina~  -2.59    0.089  -29.0    0      -2.77   -2.42
##  6 Mislabeling model_nameFaster~  -2.75    0.095  -28.8    0      -2.94   -2.57
##  7 Mislabeling model_nameCascad~  -2.26    0.078  -28.9    0      -2.42   -2.11
##  8 Untargeted  model_nameSSD       0.782   0.058   13.4    0       0.668   0.896
##  9 Untargeted  model_nameRetina~  -0.239   0.069   -3.46   0.001  -0.375  -0.104
## 10 Untargeted  model_nameCascad~  -0.505   0.074   -6.85   0      -0.65   -0.361
## # ... with abbreviated variable names 1: estimate, 2: std.error, 3: statistic,
## #   4: conf.low, 5: conf.high
```

```
cap <- table_caption("detection models, split by attack,", "All attacks, especially vanishing and mislab

print_statistics(reg_est, cap)
```

Table 1: We run a logistic model regressing success against detection models, split by attack, in the randomized attack experiment. All attacks, especially vanishing and mislabeling, obtain higher success on 1-stage (YOLOv3, SSD) than 2-stage (Faster R-CNN, Cascade R-CNN) detectors. However, the 1-stage RetinaNet is as resilient as 2-stage detectors. Table headers are explained in Appendix **??**.

Group	Regression							
Attack	term	sig	estimate	std.error	statistic	p.value	conf.low	conf.high
Vanishing	YOLOv3		0.000					
	SSD		0.041	0.044	0.924	0.355	-0.046	0.127
	RetinaNet	*	-1.492	0.060	-24.865	0.000	-1.610	-1.375
	Faster R-CNN	*	-2.161	0.075	-28.651	0.000	-2.311	-2.015
	Cascade R-CNN	*	-1.788	0.066	-27.097	0.000	-1.919	-1.660
	YOLOv3		0.000					

Mislabeling	SSD	*	-0.283	0.046	-6.183	0.000	-0.372	-0.193
	RetinaNet	*	-2.594	0.089	-29.029	0.000	-2.773	-2.422
	Faster R-CNN	*	-2.752	0.095	-28.826	0.000	-2.944	-2.569
	Cascade R-CNN	*	-2.259	0.078	-28.907	0.000	-2.415	-2.109
Untargeted	YOLOv3		0.000					
	SSD	*	0.782	0.058	13.411	0.000	0.668	0.896
	RetinaNet	*	-0.239	0.069	-3.463	0.001	-0.375	-0.104
	Faster R-CNN		-0.031	0.066	-0.462	0.644	-0.160	0.099
	Cascade R-CNN	*	-0.505	0.074	-6.850	0.000	-0.650	-0.361

```
# compare attacks against vanishing
# grouped by models
model <- partial(glm_model, predictor = "loss_target")

reg_est <- get_tidied_reg(
  model, data, model_name
)
```

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

```
ext_sig(reg_est)
```

```
## Total 15 predictors:
## 7 (47%) significant;
## 7 (47%) both
```

```
## # A tibble: 7 x 8
## # Groups:   model_name [5]
##   model_name    term              estim~1 std.e~2 stati~3 p.value conf.~4 conf.~5
##   <fct>         <chr>               <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 YOLOv3        loss_targetUnta~  -1.25    0.056  -22.3         0  -1.36   -1.14
## 2 SSD           loss_targetMisl~  -0.318   0.046   -6.99        0  -0.408  -0.229
## 3 SSD           loss_targetUnta~  -0.509   0.047  -10.8         0  -0.601  -0.417
## 4 RetinaNet     loss_targetMisl~  -1.10    0.098  -11.2         0  -1.29   -0.907
## 5 Faster R-CNN  loss_targetMisl~  -0.586   0.113   -5.17        0  -0.811  -0.366
## 6 Faster R-CNN  loss_targetUnta~   0.881   0.083   10.6         0   0.719   1.04
## 7 Cascade R-CNN loss_targetMisl~  -0.466   0.092   -5.06        0  -0.648  -0.287
## # ... with abbreviated variable names 1: estimate, 2: std.error, 3: statistic,
## #   4: conf.low, 5: conf.high
```

```
cap <- table_caption("attacks, split by detection models", "Targeted attacks achieve higher success thar

print_statistics(reg_est, cap)
```

Table 2: We run a logistic model regressing success against attacks, split by detection models in the randomized attack experiment. Targeted attacks achieve higher success than untargeted attack on YOLOv3 and SSD. Within targeted attacks, vanishing attacks achieve higher success than mislabeling attack, except on YOLOv3. Table headers are explained in Appendix **??**.

Group	Regression

Model	term	sig	estimate	std.error	statistic	p.value	conf.low	conf.high
YOLOv3	Vanishing		0.000					
	Mislabeling		0.005	0.044	0.110	0.912	-0.082	0.091
	Untargeted	*	-1.250	0.056	-22.340	0.000	-1.360	-1.141
SSD	Vanishing		0.000					
	Mislabeling	*	-0.318	0.046	-6.990	0.000	-0.408	-0.229
	Untargeted	*	-0.509	0.047	-10.844	0.000	-0.601	-0.417
RetinaNet	Vanishing		0.000					
	Mislabeling	*	-1.097	0.098	-11.181	0.000	-1.292	-0.907
	Untargeted		0.003	0.072	0.036	0.971	-0.139	0.145
Faster R-CNN	Vanishing		0.000					
	Mislabeling	*	-0.586	0.113	-5.171	0.000	-0.811	-0.366
	Untargeted	*	0.881	0.083	10.587	0.000	0.719	1.045
Cascade R-CNN	Vanishing		0.000					
	Mislabeling	*	-0.466	0.092	-5.056	0.000	-0.648	-0.287
	Untargeted		0.033	0.082	0.408	0.683	-0.127	0.193

```
# num_iteration
reg_est <- get_tidied_reg(
  partial(glm_model, predictor = "log(num_iteration)"),
  success_expanded_data,
)
```

```
## `summarise()` has grouped output by 'model_name', 'loss_target'. You can
## override using the `.groups` argument.
```

```
ext_sig(reg_est, "pos")
```

```
## Total 15 predictors:
## 15 (100%) significant;
## 15 (100%) pos
```

```
## # A tibble: 15 x 9
## # Groups:   model_name, loss_target [15]
##    model_name    loss_ta~1 term  estim~2 std.e~3 stati~4 p.value conf.~5 conf.~6
##    <ord>         <ord>     <chr>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1 YOLOv3        Vanishing log(~   0.48    0.018   26.7        0   0.445   0.515
##  2 YOLOv3        Mislabel~ log(~   0.397   0.017   23.3        0   0.363   0.43
##  3 YOLOv3        Untarget~ log(~   0.174   0.023    7.40       0   0.128   0.22
##  4 SSD           Vanishing log(~   0.528   0.018   29.0        0   0.493   0.564
##  5 SSD           Mislabel~ log(~   0.452   0.019   23.4        0   0.414   0.49
##  6 SSD           Untarget~ log(~   0.246   0.018   13.7        0   0.211   0.281
##  7 RetinaNet     Vanishing log(~   0.475   0.033   14.3        0   0.411   0.541
##  8 RetinaNet     Mislabel~ log(~   0.312   0.048    6.49       0   0.219   0.408
##  9 RetinaNet     Untarget~ log(~   0.328   0.029   11.2        0   0.271   0.385
## 10 Faster R-CNN  Vanishing log(~   0.394   0.042    9.32       0   0.313   0.479
## 11 Faster R-CNN  Mislabel~ log(~   0.264   0.051    5.20       0   0.166   0.364
## 12 Faster R-CNN  Untarget~ log(~   0.44    0.03    14.5        0   0.381   0.5
## 13 Cascade R-CNN Vanishing log(~   0.495   0.039   12.8        0   0.42    0.572
## 14 Cascade R-CNN Mislabel~ log(~   0.327   0.042    7.76       0   0.245   0.41
```

```
## 15 Cascade R-CNN Untarget~ log(~   0.291   0.033    8.89       0   0.228   0.356
## # ... with abbreviated variable names 1: loss_target, 2: estimate,
## #   3: std.error, 4: statistic, 5: conf.low, 6: conf.high
```

```
cap <- table_caption(glue("log({itr_lab})"), "Success rates increase with attack iterations for all mod

print_statistics(reg_est, cap)
```

Table 3: We run a logistic model regressing success against log(attack iterations) in the randomized attack experiment. Success rates increase with attack iterations for all models and attacks. Table headers are explained in Appendix **??**.

Group	Regression							
Attack	term	sig	estimate	std.error	statistic	p.value	conf.low	conf.high
YOLOv3								
Vanishing	log(iterations)	*	0.480	0.018	26.729	0	0.445	0.515
Mislabeling	log(iterations)	*	0.397	0.017	23.267	0	0.363	0.430
Untargeted	log(iterations)	*	0.174	0.023	7.404	0	0.128	0.220
SSD								
Vanishing	log(iterations)	*	0.528	0.018	29.009	0	0.493	0.564
Mislabeling	log(iterations)	*	0.452	0.019	23.386	0	0.414	0.490
Untargeted	log(iterations)	*	0.246	0.018	13.735	0	0.211	0.281
RetinaNet								
Vanishing	log(iterations)	*	0.475	0.033	14.339	0	0.411	0.541
Mislabeling	log(iterations)	*	0.312	0.048	6.489	0	0.219	0.408
Untargeted	log(iterations)	*	0.328	0.029	11.206	0	0.271	0.385
Faster R-CNN								
Vanishing	log(iterations)	*	0.394	0.042	9.316	0	0.313	0.479
Mislabeling	log(iterations)	*	0.264	0.051	5.204	0	0.166	0.364
Untargeted	log(iterations)	*	0.440	0.030	14.511	0	0.381	0.500
Cascade R-CNN								
Vanishing	log(iterations)	*	0.495	0.039	12.772	0	0.420	0.572
Mislabeling	log(iterations)	*	0.327	0.042	7.758	0	0.245	0.410
Untargeted	log(iterations)	*	0.291	0.033	8.886	0	0.228	0.356

```
# cache.lazy = FALSE needed to avoid errors with large bbox .parquets
attack_bbox <- "ground_truth"

bbox_fnames <-
  dir_ls(data_dir, glob = glue("*{params$simulation}*_bboxes.parquet"))

# Every bbox whether ground-truth, predicted or attacked is a row and the columns are the sample and bb
bbox_raw_data <- get_data(bbox_fnames, read_parquet) |>
  glimpse() |>
  lazy_dt()
```

```
## Rows: 6,449,532
## Columns: 71
```

```
## $ fname                          <chr> "./data/random/results/random_repeat_10~
## $ sample_id                      <chr> "63baef4898dbbc7a545ee3a0", "63baef4898~
## $ sample_path                    <chr> "/projectsp/f_ps848_1/zhaobin/adversari~
## $ sample_width                   <int> 640, 640, 640, 640, 640, 640, 640, 640,~
## $ sample_height                  <int> 480, 480, 480, 480, 480, 480, 480, 480,~
## $ sample_mislabel_class_10       <chr> "truck", "truck", "truck", "truck", "tr~
## $ sample_mislabel_proba_10       <dbl> 1.266122e-03, 1.266122e-03, 1.266122e-0~
## $ sample_mislabel_class_50       <chr> "truck", "truck", "truck", "truck", "tr~
## $ sample_mislabel_proba_50       <dbl> 1.266122e-03, 1.266122e-03, 1.266122e-0~
## $ sample_mislabel_class_100      <chr> "truck", "truck", "truck", "truck", "tr~
## $ sample_mislabel_proba_100      <dbl> 1.266122e-03, 1.266122e-03, 1.266122e-0~
## $ sample_mislabel_class_200      <chr> "truck", "truck", "truck", "truck", "tr~
## $ sample_mislabel_proba_200      <dbl> 1.266122e-03, 1.266122e-03, 1.266122e-0~
## $ sample_mislabel_intended_200   <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_mislabel_50             <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_success_10              <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_mislabel_intended_50    <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_mislabel_200            <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_vanish_10               <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_success_100             <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_mislabel_intended_100   <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_success_50              <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_mislabel_100            <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_attack                  <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRU~
## $ sample_success_200             <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ bbox_id                        <chr> "63baef4898dbbc7a545ee381", "63baef4898~
## $ bbox_class                     <chr> "car", "bus", "car", "bus", "car", "bus~
## $ bbox_xywhn                     <list<double>> <0.8419063, 0.3694167, 0.14742~
## $ bbox_conf                      <dbl> NA, NA, 0.6037431, 0.9982775, 0.9977509~
## $ bbox_res_eval                  <chr> "tp", "tp", "tp", "tp", NA, NA, NA, NA,~
## $ bbox_iou_eval                  <dbl> 0.6468383, 0.9772157, 0.6468383, 0.9772~
## $ bbox_res_pgd_100_eval          <chr> "tp", "tp", NA, NA, NA, NA, NA, NA, "tp~
## $ bbox_iou_pgd_100_eval          <dbl> 0.9688864, 0.9713300, NA, NA, NA, NA, N~
## $ bbox_res_pgd_100_mislabel_eval <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_100_mislabel_eval <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_pgd_10_eval           <chr> "tp", "tp", NA, NA, "tp", "tp", NA, NA,~
## $ bbox_iou_pgd_10_eval           <dbl> 0.9349849, 0.9773309, NA, NA, 0.9349849~
## $ bbox_res_pgd_10_mislabel_eval  <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_10_mislabel_eval  <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_pgd_200_eval          <chr> "tp", "tp", NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_200_eval          <dbl> 0.9609979, 0.9730092, NA, NA, NA, NA, N~
## $ bbox_res_pgd_200_mislabel_eval <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_200_mislabel_eval <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_pgd_50_eval           <chr> "tp", "tp", NA, NA, NA, NA, "tp", "tp",~
## $ bbox_iou_pgd_50_eval           <dbl> 0.9551206, 0.9793539, NA, NA, NA, NA, 0~
## $ bbox_res_pgd_50_mislabel_eval  <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_50_mislabel_eval  <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_predictions_eval      <chr> "tp", "tp", "tp", "tp", NA, NA, NA, NA,~
## $ bbox_iou_predictions_eval      <dbl> 0.6468383, 0.9772157, 0.6468383, 0.9772~
## $ bbox_target_100                <lgl> FALSE, TRUE, FALSE, TRUE, NA, NA, NA, N~
## $ bbox_perturb_10                <lgl> TRUE, FALSE, TRUE, FALSE, NA, NA, NA, N~
## $ bbox_target_200                <lgl> FALSE, TRUE, FALSE, TRUE, NA, NA, NA, N~
## $ bbox_perturb_200               <lgl> TRUE, FALSE, TRUE, FALSE, NA, NA, NA, N~
## $ bbox_perturb_50                <lgl> TRUE, FALSE, TRUE, FALSE, NA, NA, NA, N~
```

```
## $ bbox_target_10                <lgl> FALSE, TRUE, FALSE, TRUE, NA, NA, NA, N~
## $ bbox_target_50                <lgl> FALSE, TRUE, FALSE, TRUE, NA, NA, NA, N~
## $ bbox_perturb_100              <lgl> TRUE, FALSE, TRUE, FALSE, NA, NA, NA, N~
## $ bbox_type                     <chr> "ground_truth", "ground_truth", "predic~
## $ bbox_mislabel_50              <lgl> NA, NA, NA, NA, NA, NA, FALSE, FALSE, N~
## $ bbox_mislabel_100             <lgl> NA, NA, NA, NA, NA, NA, NA, NA, FALSE, ~
## $ bbox_mislabel_200             <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ model_name                    <ord> Cascade R-CNN, Cascade R-CNN, Cascade R~
## $ loss_target                   <ord> Mislabeling, Mislabeling, Mislabeling, ~
## $ attack_bbox                   <chr> "ground_truth", "ground_truth", "ground~
## $ perturb_fun                   <chr> "perturb_inside", "perturb_inside", "pe~
## $ sample_vanish_50              <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ sample_vanish_200             <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ sample_mislabel_10            <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ sample_mislabel_intended_10   <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ sample_vanish_100             <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_mislabel_10              <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
```

```
# check whether target and perturb bboxes and
# mislabel classes are seeded across iterations
cols_start_equal(bbox_raw_data, c(
  "bbox_target", "bbox_perturb",
  "sample_mislabel_class", "sample_mislabel_proba"
))
```

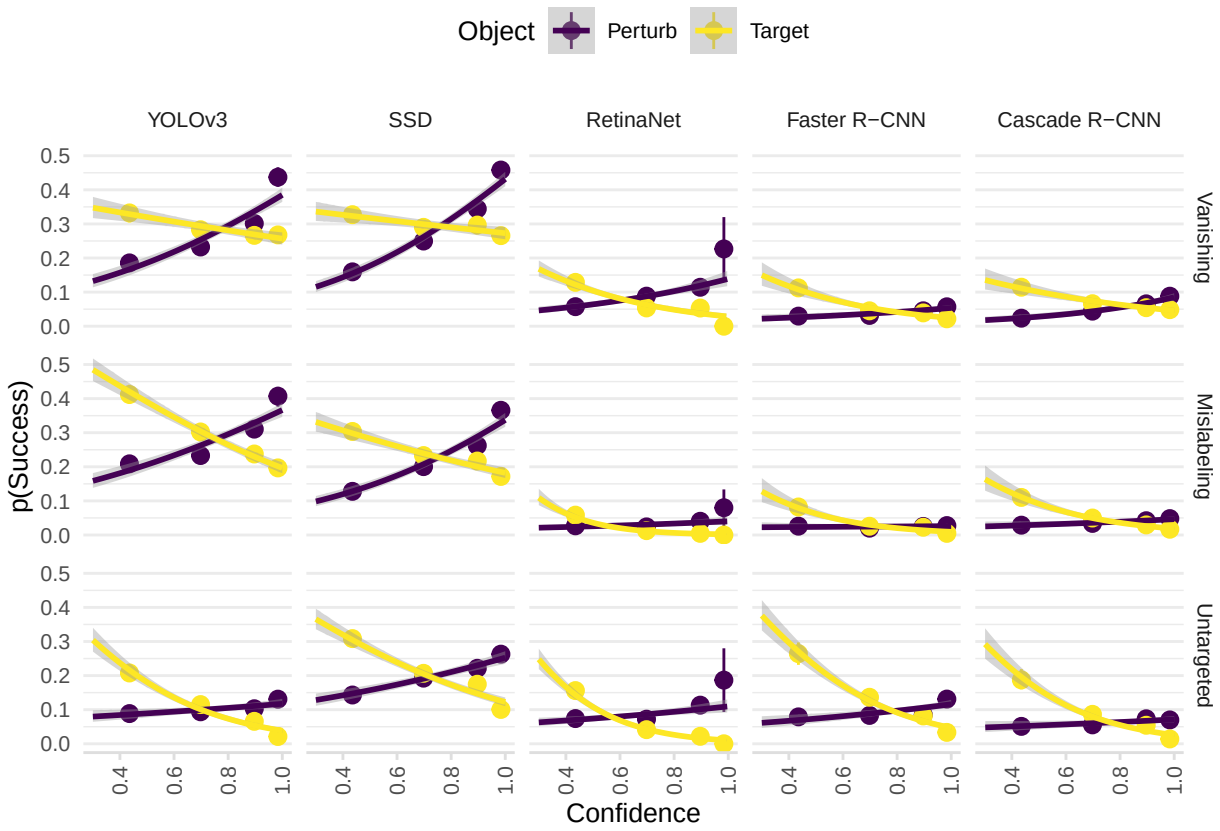
```
## Columns starting with `bbox_target` are equal: TRUE
## Columns starting with `bbox_perturb` are equal: TRUE
## Columns starting with `sample_mislabel_class` are equal: TRUE
## Columns starting with `sample_mislabel_proba` are equal: TRUE
```

```
# bbox confidence always based on predicted bbox
bbox_conf_data <- bbox_raw_data |>
  filter(bbox_type == "predictions") |>
  wrangle_success() |>
  glimpse()
```

```
## Rows: 150,000
## Columns: 63
## $ fname                         <chr> "./data/random/results/random_repeat_10~
## $ sample_id                     <chr> "63baef4898dbbc7a545ee3a0", "63baef4898~
## $ sample_path                   <chr> "/projectsp/f_ps848_1/zhaobin/adversari~
## $ sample_width                  <int> 640, 425, 640, 416, 429, 385, 480, 478,~
## $ sample_height                 <int> 480, 640, 427, 640, 640, 308, 640, 640,~
## $ sample_mislabel_class_10      <chr> "truck", "train", "dog", "dog", "fire h~
## $ sample_mislabel_proba_10      <dbl> 1.266122e-03, 1.629095e-04, 3.693961e-0~
## $ sample_mislabel_class_50      <chr> "truck", "train", "dog", "dog", "fire h~
## $ sample_mislabel_proba_50      <dbl> 1.266122e-03, 1.629095e-04, 3.693961e-0~
## $ sample_mislabel_class_100     <chr> "truck", "train", "dog", "dog", "fire h~
## $ sample_mislabel_proba_100     <dbl> 1.266122e-03, 1.629095e-04, 3.693961e-0~
## $ sample_mislabel_class_200     <chr> "truck", "train", "dog", "dog", "fire h~
## $ sample_mislabel_proba_200     <dbl> 1.266122e-03, 1.629095e-04, 3.693961e-0~
## $ sample_mislabel_intended_200  <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_mislabel_50            <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_mislabel_intended_50   <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_mislabel_200           <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_vanish_10              <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
```

```
## $ sample_mislabel_intended_100   <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_mislabel_100            <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_attack                  <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRU~
## $ bbox_id                        <chr> "63baef4e98dbbc7a545f1fe8", "63baef4e98~
## $ bbox_class                     <chr> "bus", "clock", "person", "teddy bear",~
## $ bbox_xywhn                     <list<double>> <0.05912231, 0.22379084, 0.761~
## $ bbox_conf                      <dbl> 0.9982775, 0.9760631, 0.8126031, 0.7020~
## $ bbox_res_eval                  <chr> "tp", "tp", "tp", "tp", "fp", "tp", "tp~
## $ bbox_iou_eval                  <dbl> 0.9772157, 0.7299863, 0.9619440, 0.7784~
## $ bbox_res_pgd_100_eval          <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_100_eval          <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_pgd_100_mislabel_eval <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_100_mislabel_eval <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_pgd_10_eval           <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_10_eval           <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_pgd_10_mislabel_eval  <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_10_mislabel_eval  <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_pgd_200_eval          <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_200_eval          <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_pgd_200_mislabel_eval <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_200_mislabel_eval <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_pgd_50_eval           <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_50_eval           <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_pgd_50_mislabel_eval  <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_50_mislabel_eval  <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_predictions_eval      <chr> "tp", "tp", "tp", "tp", "tp", "tp", "tp~
## $ bbox_iou_predictions_eval      <dbl> 0.9772157, 0.7299863, 0.9619440, 0.7784~
## $ bbox_type                      <chr> "predictions", "predictions", "predicti~
## $ bbox_mislabel_50               <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_mislabel_100              <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_mislabel_200              <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ model_name                     <ord> Cascade R-CNN, Cascade R-CNN, Cascade R~
## $ loss_target                    <ord> Mislabeling, Mislabeling, Mislabeling, ~
## $ attack_bbox                    <chr> "ground_truth", "ground_truth", "ground~
## $ perturb_fun                    <chr> "perturb_inside", "perturb_inside", "pe~
## $ sample_vanish_50               <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ sample_vanish_200              <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ sample_mislabel_10             <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ sample_mislabel_intended_10    <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ sample_vanish_100              <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_mislabel_10               <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ target_or_perturb_boolean      <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRU~
## $ target_or_perturb              <ord> Target, Target, Target, Target, Target,~
## $ attack_itr                     <int> 200, 200, 200, 200, 200, 200, 200, 200,~
## $ success                        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
```

```
bbox_conf_data |>
  graph_attr(bbox_conf, "Confidence")
```




```
# restrict to target
pred_name <- "target confidence"
main_pt <- glue("Lower {pred_name} significantly increases success rates for all models and attacks")

bbox_conf_graph <- bbox_conf_data |> filter(target_or_perturb == "Target")
bbox_conf_graph |>
  graph_attr(bbox_conf, pred_name)
```

```
model <- partial(glm_model, predictor = "bbox_conf")
data <- bbox_conf_graph

reg_est <- get_tidied_reg(model, data)
```

```
## `summarise()` has grouped output by 'model_name', 'loss_target'. You can
## override using the `.groups` argument.
```

```
ext_sig(reg_est, "neg")
```

```
## Total 15 predictors:
## 15 (100%) significant;
## 15 (100%) neg
```

```
## # A tibble: 15 x 9
## # Groups:   model_name, loss_target [15]
##    model_name    loss_ta~1 term  estim~2 std.e~3 stati~4 p.value conf.~5 conf.~6
##    <ord>         <ord>     <chr>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1 YOLOv3        Vanishing bbox~  -0.618   0.141   -4.38   0      -0.895  -0.341
##  2 YOLOv3        Mislabel~ bbox~  -1.92    0.142  -13.5    0      -2.20   -1.64
##  3 YOLOv3        Untarget~ bbox~  -3.42    0.215  -15.9    0      -3.84   -3.00
```

```
##  4 SSD           Vanishing bbox~  -0.428   0.13    -3.29   0.001  -0.684  -0.173
##  5 SSD           Mislabel~ bbox~  -1.14    0.14    -8.20   0      -1.42   -0.871
##  6 SSD           Untarget~ bbox~  -2.02    0.149  -13.6    0      -2.32   -1.73
##  7 RetinaNet     Vanishing bbox~  -2.76    0.278   -9.93   0      -3.31   -2.22
##  8 RetinaNet     Mislabel~ bbox~  -5.95    0.595  -10.0    0      -7.16   -4.83
##  9 RetinaNet     Untarget~ bbox~  -5.00    0.328  -15.2    0      -5.66   -4.37
## 10 Faster R-CNN  Vanishing bbox~  -2.81    0.29    -9.71   0      -3.38   -2.24
## 11 Faster R-CNN  Mislabel~ bbox~  -3.93    0.382  -10.3    0      -4.68   -3.18
## 12 Faster R-CNN  Untarget~ bbox~  -3.58    0.207  -17.3    0      -3.98   -3.17
## 13 Cascade R-CNN Vanishing bbox~  -1.69    0.259   -6.53   0      -2.19   -1.18
## 14 Cascade R-CNN Mislabel~ bbox~  -3.33    0.305  -10.9    0      -3.93   -2.73
## 15 Cascade R-CNN Untarget~ bbox~  -3.93    0.25   -15.7    0      -4.42   -3.44
## # ... with abbreviated variable names 1: loss_target, 2: estimate,
## #   3: std.error, 4: statistic, 5: conf.low, 6: conf.high
```

```
print_statistics(reg_est, table_caption(pred_name, main_pt))
```

Table 4: We run a logistic model regressing success against target confidence in the randomized attack experiment. Lower target confidence significantly increases success rates for all models and attacks. Table headers are explained in Appendix **??**.

Group	Regression							
Attack	term	sig	estimate	std.error	statistic	p.value	conf.low	conf.high
YOLOv3								
Vanishing	confidence	*	-0.618	0.141	-4.375	0.000	-0.895	-0.341
Mislabeling	confidence	*	-1.924	0.142	-13.520	0.000	-2.203	-1.645
Untargeted	confidence	*	-3.417	0.215	-15.919	0.000	-3.841	-2.999
SSD								
Vanishing	confidence	*	-0.428	0.130	-3.288	0.001	-0.684	-0.173
Mislabeling	confidence	*	-1.144	0.140	-8.199	0.000	-1.418	-0.871
Untargeted	confidence	*	-2.024	0.149	-13.602	0.000	-2.317	-1.733
RetinaNet								
Vanishing	confidence	*	-2.762	0.278	-9.925	0.000	-3.314	-2.222
Mislabeling	confidence	*	-5.951	0.595	-10.002	0.000	-7.162	-4.826
Untargeted	confidence	*	-5.002	0.328	-15.238	0.000	-5.657	-4.370
Faster R-CNN								
Vanishing	confidence	*	-2.814	0.290	-9.706	0.000	-3.382	-2.244
Mislabeling	confidence	*	-3.927	0.382	-10.290	0.000	-4.683	-3.184
Untargeted	confidence	*	-3.578	0.207	-17.308	0.000	-3.985	-3.174
Cascade R-CNN								
Vanishing	confidence	*	-1.690	0.259	-6.533	0.000	-2.194	-1.179
Mislabeling	confidence	*	-3.329	0.305	-10.928	0.000	-3.928	-2.732
Untargeted	confidence	*	-3.927	0.250	-15.679	0.000	-4.421	-3.438

```
perturb_error_data <- bbox_conf_data |>
  filter(target_or_perturb == "Perturb") |>
  group_by(model_name, loss_target) |>
```

```
  summarise(perturb_error = 1 - mean(success)) |>
  glimpse()
```

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

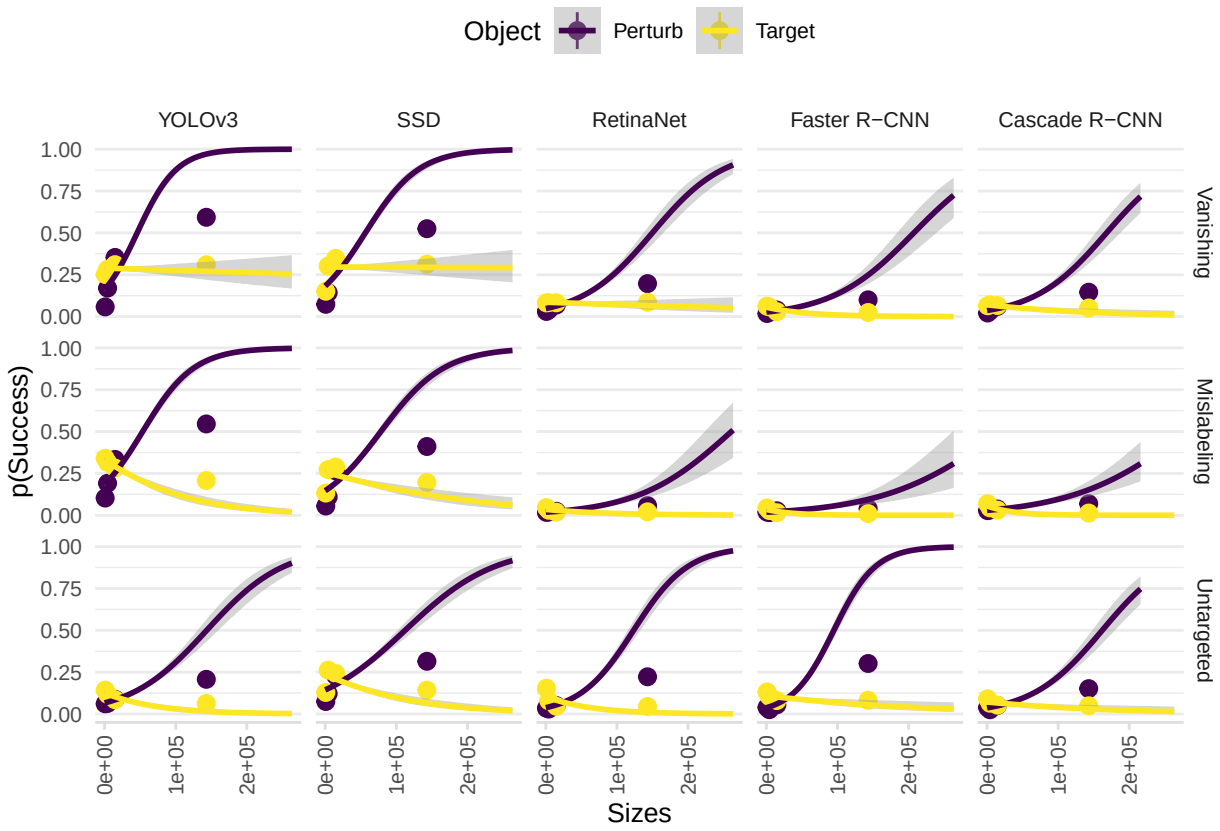
```
## Rows: 15
## Columns: 3
## Groups: model_name [5]
## $ model_name    <ord> YOLOv3, YOLOv3, YOLOv3, SSD, SSD, SSD, RetinaNet, Retina~
## $ loss_target   <ord> Vanishing, Mislabeling, Untargeted, Vanishing, Mislabeli~
## $ perturb_error <dbl> 0.7126, 0.7116, 0.8964, 0.7042, 0.7660, 0.7984, 0.9168, ~
```

```
# bbox sizes typically based on ground-truth attacked bbox
# not applicable to "arbitrary" attack since the bbox sizes are set experimentally
# regression with distances later
bbox_size_data <- bbox_raw_data |>
  filter(bbox_type == attack_bbox) |>
  wrangle_success() |>
  # hoist not implemented in dtplyr
  as_tibble() |>
  # bbox_xywhn == normalized x1, y1, w, h
  hoist(bbox_xywhn, bbox_xn = 1, bbox_yn = 2, bbox_wn = 3, bbox_hn = 4) |>
  mutate(
    bbox_w = bbox_wn * sample_width,
    bbox_h = bbox_hn * sample_height,
    bbox_size = bbox_w * bbox_h,
  ) |>
  glimpse()
```

```
## Rows: 150,000
## Columns: 69
## $ fname                          <chr> "./data/random/results/random_repeat_10~
## $ sample_id                      <chr> "63baef4898dbbc7a545ee3a0", "63baef4898~
## $ sample_path                    <chr> "/projectsp/f_ps848_1/zhaobin/adversari~
## $ sample_width                   <int> 640, 425, 640, 416, 429, 385, 480, 478,~
## $ sample_height                  <int> 480, 640, 427, 640, 640, 308, 640, 640,~
## $ sample_mislabel_class_10       <chr> "truck", "train", "dog", "dog", "fire h~
## $ sample_mislabel_proba_10       <dbl> 1.266122e-03, 1.629095e-04, 3.693961e-0~
## $ sample_mislabel_class_50       <chr> "truck", "train", "dog", "dog", "fire h~
## $ sample_mislabel_proba_50       <dbl> 1.266122e-03, 1.629095e-04, 3.693961e-0~
## $ sample_mislabel_class_100      <chr> "truck", "train", "dog", "dog", "fire h~
## $ sample_mislabel_proba_100      <dbl> 1.266122e-03, 1.629095e-04, 3.693961e-0~
## $ sample_mislabel_class_200      <chr> "truck", "train", "dog", "dog", "fire h~
## $ sample_mislabel_proba_200      <dbl> 1.266122e-03, 1.629095e-04, 3.693961e-0~
## $ sample_mislabel_intended_200   <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_mislabel_50             <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_mislabel_intended_50    <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_mislabel_200            <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_vanish_10               <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_mislabel_intended_100   <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_mislabel_100            <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_attack                  <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRU~
## $ bbox_id                        <chr> "63baef4898dbbc7a545ee382", "63baef4898~
## $ bbox_class                     <chr> "bus", "clock", "person", "teddy bear",~
```

```
## $ bbox_xn                          <dbl> 0.04898438, 0.23800000, 0.78331250, 0.1~
## $ bbox_yn                          <dbl> 0.22185417, 0.23660938, 0.34098361, 0.5~
## $ bbox_wn                          <dbl> 0.77534375, 0.09374118, 0.06617188, 0.2~
## $ bbox_hn                          <dbl> 0.51125000, 0.12831250, 0.17384075, 0.2~
## $ bbox_conf                        <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_eval                    <chr> "tp", "tp", "tp", "tp", "fn", "tp", "tp~
## $ bbox_iou_eval                    <dbl> 0.9772157, 0.7299863, 0.9619440, 0.7784~
## $ bbox_res_pgd_100_eval            <chr> "tp", "tp", "tp", "tp", "tp", "tp", "tp~
## $ bbox_iou_pgd_100_eval            <dbl> 0.9713300, 0.7300485, 0.9619440, 0.7784~
## $ bbox_res_pgd_100_mislabel_eval   <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_100_mislabel_eval   <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_pgd_10_eval             <chr> "tp", "tp", "tp", "tp", "tp", "tp", "tp~
## $ bbox_iou_pgd_10_eval             <dbl> 0.9773309, 0.7300411, 0.9619440, 0.7784~
## $ bbox_res_pgd_10_mislabel_eval    <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_10_mislabel_eval    <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_pgd_200_eval            <chr> "tp", "tp", "tp", "tp", "tp", "tp", "tp~
## $ bbox_iou_pgd_200_eval            <dbl> 0.9730092, 0.7300449, 0.9619440, 0.7784~
## $ bbox_res_pgd_200_mislabel_eval   <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_200_mislabel_eval   <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_pgd_50_eval             <chr> "tp", "tp", "tp", "tp", "tp", "tp", "tp~
## $ bbox_iou_pgd_50_eval             <dbl> 0.9793539, 0.7300653, 0.9619440, 0.7784~
## $ bbox_res_pgd_50_mislabel_eval    <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_50_mislabel_eval    <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_predictions_eval        <chr> "tp", "tp", "tp", "tp", "tp", "tp", "tp~
## $ bbox_iou_predictions_eval        <dbl> 0.9772157, 0.7299863, 0.9619440, 0.7784~
## $ bbox_type                        <chr> "ground_truth", "ground_truth", "ground~
## $ bbox_mislabel_50                 <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_mislabel_100                <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_mislabel_200                <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ model_name                       <ord> Cascade R-CNN, Cascade R-CNN, Cascade R~
## $ loss_target                      <ord> Mislabeling, Mislabeling, Mislabeling, ~
## $ attack_bbox                      <chr> "ground_truth", "ground_truth", "ground~
## $ perturb_fun                      <chr> "perturb_inside", "perturb_inside", "pe~
## $ sample_vanish_50                 <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ sample_vanish_200                <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ sample_mislabel_10               <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ sample_mislabel_intended_10      <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ sample_vanish_100                <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_mislabel_10                 <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ target_or_perturb_boolean        <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRU~
## $ target_or_perturb                <ord> Target, Target, Target, Target, Target,~
## $ attack_itr                       <int> 200, 200, 200, 200, 200, 200, 200, 200,~
## $ success                          <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ bbox_w                           <dbl> 496.22, 39.84, 42.35, 95.10, 4.48, 125.~
## $ bbox_h                           <dbl> 245.40, 82.12, 74.23, 162.50, 10.47, 19~
## $ bbox_size                        <dbl> 121772.3880, 3271.6608, 3143.6405, 1545~
```

```
bbox_size_data |>
  graph_attr(bbox_size, "Sizes")
```




```
# bbox distances typically based on ground-truth attacked bbox as in sizes
# not applicable to "arbitrary" attack since the bbox distances are set experimentally
# regression with sizes later
bbox_dist_data <- bbox_size_data |>
  mutate(
    bbox_x1 = bbox_xn * sample_width,
    bbox_y1 = bbox_yn * sample_height,
    bbox_x2 = bbox_x1 + bbox_w,
    bbox_y2 = bbox_y1 + bbox_h,
    target_or_perturb_lower = str_to_lower(target_or_perturb)
  ) |>
  # mainly "group" by sample_id and attack iteration
  # with target bbox on one row and perturb on another
  # success, model_name, loss_target are sample attributes
  # duplicated across bboxes
  pivot_wider(
    id_cols = c(fname, sample_id, attack_itr, success, model_name, loss_target), names_from = target_or
    values_from = c(bbox_x1, bbox_y1, bbox_x2, bbox_y2, bbox_size)
  ) |>
  rowwise() |>
  mutate(bbox_dist = get_min_distance(
    bbox_x1_perturb, bbox_y1_perturb, bbox_x2_perturb, bbox_y2_perturb,
    bbox_x1_target, bbox_y1_target, bbox_x2_target, bbox_y2_target
  )) |>
  ungroup() |>
  glimpse()
```

```
## Rows: 75,000
```

```
## Columns: 17
## $ fname             <chr> "./data/random/results/random_repeat_10_mislabel_bbo~
## $ sample_id         <chr> "63baef4898dbbc7a545ee3a0", "63baef4898dbbc7a545ee3a~
## $ attack_itr        <int> 200, 200, 200, 200, 200, 200, 200, 200, 200, 200, 20~
## $ success           <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ model_name        <ord> Cascade R-CNN, Cascade R-CNN, Cascade R-CNN, Cascade~
## $ loss_target       <ord> Mislabeling, Mislabeling, Mislabeling, Mislabeling, ~
## $ bbox_x1_target    <dbl> 31.35, 101.15, 501.32, 48.25, 38.62, 14.36, 285.26, ~
## $ bbox_x1_perturb   <dbl> 538.82, 215.01, 220.59, 256.00, 104.36, 165.60, 241.~
## $ bbox_y1_target    <dbl> 106.49, 151.43, 145.60, 372.79, 213.35, 8.46, 301.63~
## $ bbox_y1_perturb   <dbl> 177.32, 139.48, 198.22, 17.26, 144.98, 157.36, 301.8~
## $ bbox_x2_target    <dbl> 527.57, 140.99, 543.67, 143.35, 43.10, 139.64, 318.5~
## $ bbox_x2_perturb   <dbl> 633.17, 276.68, 234.03, 378.25, 262.76, 350.40, 273.~
## $ bbox_y2_target    <dbl> 351.89, 233.55, 219.83, 535.29, 223.82, 205.55, 310.~
## $ bbox_y2_perturb   <dbl> 227.19, 201.15, 242.51, 257.44, 420.64, 269.36, 363.~
## $ bbox_size_target  <dbl> 121772.3880, 3271.6608, 3143.6405, 15453.7500, 46.90~
## $ bbox_size_perturb <dbl> 4705.2345, 3803.1889, 595.2576, 29362.0050, 43664.54~
## $ bbox_dist         <dbl> 11.250000, 74.020000, 267.290000, 161.231650, 61.260~
```

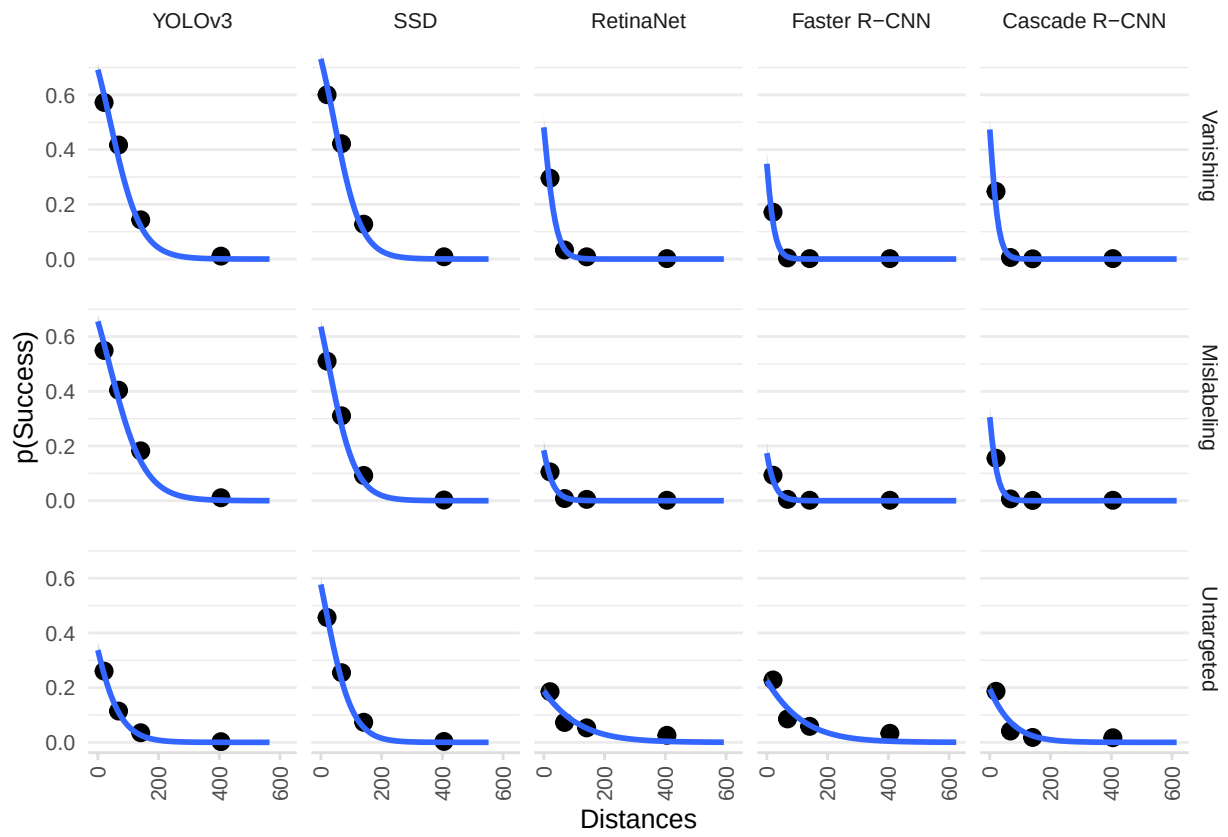
```
bbox_dist_data |>
  graph_attr(bbox_dist, "Distances")
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```




```
saveRDS(bbox_dist_data, "./analysis/rand_dist_size.RDS")
```

```
check_graph_data(bbox_dist_data, c(bbox_dist, bbox_size_perturb))

dist_lab <- "Perturb-Target Distance (100 pixels)"
size_lab <- "Perturb Box Size (100,000 squared pixels)"

pred_name <- glue("{dist_lab} and {size_lab}")
main_pt <- "Larger perturb objects significantly increase success rates for all models and attacks, exc

cap <- glue(
  "{bold_tex(main_pt)} The binned summaries",
  " graph success proportion against {str_to_lower(pred_name)} in the",
  " randomized attack experiment."
)

bbox_dist_data <- bbox_dist_data |> mutate(
  bbox_size_perturb = bbox_size_perturb / 1e5,
  bbox_dist = bbox_dist / 1e2
)

graph_dist_size <- function(g) {
  g + facet_grid(rows = vars(loss_target), cols = vars(model_name)) +
    labs(x = dist_lab, y = size_lab) +
    scale_fill_viridis_c(name = "p(Success)", breaks = c(0, .5, 1), limits = c(0, 1))
}

g <- bbox_dist_data |> ggplot(aes(bbox_dist, bbox_size_perturb, z = success)) +
```

```r
  stat_summary_2d(fun = "mean", bins = 5)

graph_dist_size(g)
```

```r
# control both
model <- partial(glm_model, predictor = "bbox_dist * bbox_size_perturb")
data <- bbox_dist_data

reg_res <- get_tidied_reg(model, data, return_mod = TRUE) |> glimpse()
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## `summarise()` has grouped output by 'model_name', 'loss_target'. You can
## override using the `.groups` argument.

## List of 2
##  $ mod   : rowws_df [15 x 4] (S3: rowwise_df/tbl_df/tbl/data.frame)
##   ..$ model_name : Ord.factor w/ 5 levels "YOLOv3"<"SSD"<..: 1 1 1 2 2 2 3 3 3 4 ...
##   ..$ loss_target: Ord.factor w/ 3 levels "Vanishing"<"Mislabeling"<..: 1 2 3 1 2 3 1 2 3 1 ...
##   ..$ data       : list<tibble[,15]> [1:15]
##   ..$ mod        :List of 15
##   ..- attr(*, "groups")= tibble [15 x 3] (S3: tbl_df/tbl/data.frame)
##  $ tidied: gropd_df [45 x 20] (S3: grouped_df/tbl_df/tbl/data.frame)
##   ..$ model_name    : Ord.factor w/ 5 levels "YOLOv3"<"SSD"<..: 1 1 1 1 1 1 1 1 1 2 ...
##   ..$ loss_target   : Ord.factor w/ 3 levels "Vanishing"<"Mislabeling"<..: 1 1 1 2 2 2 3 3 3 1 ...
##   ..$ term          : chr [1:45] "bbox_dist" "bbox_size_perturb" "bbox_dist:bbox_size_perturb" "bbox
##   ..$ variable      : chr [1:45] "bbox_dist" "bbox_size_perturb" "bbox_dist:bbox_size_perturb" "bbox
##   ..$ var_label     : Named chr [1:45] "bbox_dist" "bbox_size_perturb" "bbox_dist * bbox_size_perturb
##   .. ..- attr(*, "names")= chr [1:45] "bbox_dist" "bbox_size_perturb" "bbox_dist:bbox_size_perturb"
##   ..$ var_class     : Named chr [1:45] "numeric" "numeric" NA "numeric" ...
##   .. ..- attr(*, "names")= chr [1:45] "bbox_dist" "bbox_size_perturb" "" "bbox_dist" ...
##   ..$ var_type      : chr [1:45] "continuous" "continuous" "interaction" "continuous" ...
##   ..$ var_nlevels   : int [1:45] NA NA NA NA NA NA NA NA NA NA ...
##   ..$ contrasts     : chr [1:45] NA NA NA NA ...
##   ..$ contrasts_type: chr [1:45] NA NA NA NA ...
##   ..$ reference_row : logi [1:45] NA NA NA NA NA NA ...
##   ..$ label         : Named chr [1:45] "bbox_dist" "bbox_size_perturb" "bbox_dist * bbox_size_perturb
##   .. ..- attr(*, "names")= chr [1:45] "bbox_dist" "bbox_size_perturb" "bbox_dist:bbox_size_perturb"
##   ..$ n_obs         : Named num [1:45] 5000 5000 5000 5000 5000 5000 5000 5000 5000 5000 ...
##   .. ..- attr(*, "names")= chr [1:45] "bbox_dist" "bbox_size_perturb" "bbox_dist:bbox_size_perturb"
##   ..$ n_event       : Named num [1:45] 1437 1437 1437 1442 1442 ...
##   .. ..- attr(*, "names")= chr [1:45] "bbox_dist" "bbox_size_perturb" "bbox_dist:bbox_size_perturb"
##   ..$ estimate      : num [1:45] -1.9 6.44 -2.17 -1.71 3.18 ...
##   ..$ std.error     : num [1:45] 0.1023 0.4073 0.3442 0.0865 0.2698 ...
##   ..$ statistic     : num [1:45] -18.6 15.8 -6.3 -19.7 11.8 ...
##   ..$ p.value       : num [1:45] 3.17e-77 2.94e-56 3.04e-10 1.47e-86 4.32e-32 ...
##   ..$ conf.low      : num [1:45] -2.11 5.66 -2.85 -1.88 2.67 ...
##   ..$ conf.high     : num [1:45] -1.71 7.25 -1.5 -1.54 3.72 ...
##   ..- attr(*, "groups")= tibble [15 x 3] (S3: tbl_df/tbl/data.frame)
##   .. ..- attr(*, ".drop")= logi TRUE
```

```
reg_est <- reg_res$tidied

ext_sig(reg_est, "neg", "bbox_dist")
```

```
## ----------bbox_dist----------
```

```
## Total 15 predictors:
## 15 (100%) significant;
## 15 (100%) neg
```

```
## # A tibble: 15 x 9
## # Groups:   model_name, loss_target [15]
##    model_name    loss_ta~1 term  estim~2 std.e~3 stati~4 p.value conf.~5 conf.~6
##    <ord>         <ord>     <chr>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1 YOLOv3        Vanishing bbox~   -1.90   0.102  -18.6        0   -2.11   -1.71
##  2 YOLOv3        Mislabel~ bbox~   -1.71   0.087  -19.7        0   -1.88   -1.54
##  3 YOLOv3        Untarget~ bbox~   -2.19   0.16   -13.7        0   -2.52   -1.89
##  4 SSD           Vanishing bbox~   -2.26   0.112  -20.1        0   -2.49   -2.05
##  5 SSD           Mislabel~ bbox~   -2.20   0.121  -18.2        0   -2.44   -1.97
##  6 SSD           Untarget~ bbox~   -2.30   0.124  -18.5        0   -2.55   -2.06
##  7 RetinaNet     Vanishing bbox~   -5.13   0.374  -13.7        0   -5.89   -4.42
##  8 RetinaNet     Mislabel~ bbox~   -4.41   0.525   -8.40       0   -5.49   -3.44
##  9 RetinaNet     Untarget~ bbox~   -1.56   0.151  -10.3        0   -1.86   -1.27
## 10 Faster R-CNN  Vanishing bbox~   -6.11   0.604  -10.1        0   -7.35   -4.98
## 11 Faster R-CNN  Mislabel~ bbox~   -5.57   0.63    -8.84       0   -6.87   -4.40
## 12 Faster R-CNN  Untarget~ bbox~   -1.92   0.168  -11.5        0   -2.27   -1.61
## 13 Cascade R-CNN Vanishing bbox~   -6.97   0.573  -12.2        0   -8.14   -5.89
## 14 Cascade R-CNN Mislabel~ bbox~   -6.44   0.585  -11.0        0   -7.64   -5.34
## 15 Cascade R-CNN Untarget~ bbox~   -2.68   0.224  -12.0        0   -3.13   -2.26
## # ... with abbreviated variable names 1: loss_target, 2: estimate,
## #   3: std.error, 4: statistic, 5: conf.low, 6: conf.high
```

```
ext_sig(reg_est, "pos", "bbox_size_perturb")
```

```
## ----------bbox_size_perturb----------
## Total 15 predictors:
## 14 (93%) significant;
## 14 (93%) pos
```

```
## # A tibble: 14 x 9
## # Groups:   model_name, loss_target [14]
##    model_name    loss_ta~1 term  estim~2 std.e~3 stati~4 p.value conf.~5 conf.~6
##    <ord>         <ord>     <chr>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1 YOLOv3        Vanishing bbox~   6.44    0.407   15.8        0    5.66    7.25
##  2 YOLOv3        Mislabel~ bbox~   3.18    0.27    11.8        0    2.67    3.72
##  3 YOLOv3        Untarget~ bbox~   1.36    0.182    7.47       0    1.01    1.72
##  4 SSD           Vanishing bbox~   3.90    0.313   12.5        0    3.30    4.52
##  5 SSD           Mislabel~ bbox~   2.79    0.252   11.1        0    2.31    3.30
##  6 SSD           Untarget~ bbox~   1.28    0.189    6.80       0    0.922   1.66
##  7 RetinaNet     Vanishing bbox~   1.91    0.249    7.70       0    1.44    2.42
##  8 RetinaNet     Mislabel~ bbox~   0.92    0.248    3.70       0    0.435   1.41
##  9 RetinaNet     Untarget~ bbox~   1.38    0.174    7.93       0    1.04    1.72
## 10 Faster R-CNN  Vanishing bbox~   1.64    0.275    5.96       0    1.11    2.19
## 11 Faster R-CNN  Untarget~ bbox~   1.88    0.194    9.68       0    1.50    2.26
## 12 Cascade R-CNN Vanishing bbox~   2.17    0.282    7.69       0    1.64    2.74
## 13 Cascade R-CNN Mislabel~ bbox~   0.486   0.237    2.05    0.04    0.023   0.955
## 14 Cascade R-CNN Untarget~ bbox~   0.693   0.174    3.99       0    0.352   1.03
## # ... with abbreviated variable names 1: loss_target, 2: estimate,
## #   3: std.error, 4: statistic, 5: conf.low, 6: conf.high
```

```
ext_sig(reg_est, "both", "bbox_dist:bbox_size_perturb")
```

```
## ----------bbox_dist:bbox_size_perturb----------
## Total 15 predictors:
## 8 (53%) significant;
## 8 (53%) both
```

```
## # A tibble: 8 x 9
## # Groups:   model_name, loss_target [8]
##   model_name    loss_tar~1 term  estim~2 std.e~3 stati~4 p.value conf.~5 conf.~6
##   <ord>         <ord>      <chr>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 YOLOv3        Vanishing  bbox~  -2.17    0.344   -6.30   0      -2.85   -1.50
## 2 SSD           Vanishing  bbox~  -0.978   0.306   -3.19   0.001  -1.59   -0.389
## 3 SSD           Mislabeli~ bbox~  -0.64    0.295   -2.17   0.03   -1.24   -0.079
## 4 RetinaNet     Untargeted bbox~   1.68    0.23     7.32   0       1.24    2.14
## 5 Faster R-CNN  Mislabeli~ bbox~   2.24    0.767    2.92   0.004   0.578   3.60
## 6 Faster R-CNN  Untargeted bbox~   2.14    0.259    8.26   0       1.65    2.66
## 7 Cascade R-CNN Mislabeli~ bbox~   1.83    0.798    2.29   0.022   0.144   3.28
## 8 Cascade R-CNN Untargeted bbox~   2.18    0.258    8.44   0       1.68    2.69
## # ... with abbreviated variable names 1: loss_target, 2: estimate,
## #   3: std.error, 4: statistic, 5: conf.low, 6: conf.high
```

```
print_statistics(reg_est, table_caption(pred_name, main_pt))
```

Table 5: We run a logistic model regressing success against perturb-target distance (100 pixels) and perturb box size (100,000 squared pixels) in the randomized attack experiment. Larger perturb objects significantly increase success rates for all models and attacks, except for mislabeling attack on Faster R-CNN, after controlling for perturb-target distances; shorter perturb-target distances significantly increase success rates for all models and attacks, after controlling for perturb object sizes. Table headers are explained in Appendix **??**.

Group	Regression							
Attack	term	sig	estimate	std.error	statistic	p.value	conf.low	conf.high
YOLOv3								
Vanishing	distance	*	-1.903	0.102	-18.601	0.000	-2.107	-1.706
	size	*	6.436	0.407	15.804	0.000	5.656	7.252
	distance * size	*	-2.167	0.344	-6.297	0.000	-2.853	-1.502
Mislabeling	distance	*	-1.706	0.087	-19.719	0.000	-1.879	-1.540
	size	*	3.182	0.270	11.791	0.000	2.667	3.724
	distance * size		-0.384	0.252	-1.523	0.128	-0.886	0.102
Untargeted	distance	*	-2.191	0.160	-13.656	0.000	-2.515	-1.886
	size	*	1.357	0.182	7.470	0.000	1.007	1.720
	distance * size		0.444	0.287	1.547	0.122	-0.138	0.992
SSD								
Vanishing	distance	*	-2.264	0.112	-20.125	0.000	-2.488	-2.047
	size	*	3.896	0.313	12.467	0.000	3.299	4.524
	distance * size	*	-0.978	0.306	-3.194	0.001	-1.594	-0.389
Mislabeling	distance	*	-2.203	0.121	-18.194	0.000	-2.445	-1.970
	size	*	2.787	0.252	11.061	0.000	2.306	3.295
	distance * size	*	-0.640	0.295	-2.172	0.030	-1.238	-0.079

	Untargeted	distance	*	-2.299	0.124	-18.514	0.000	-2.547	-2.060
		size	*	1.283	0.189	6.805	0.000	0.922	1.662
		distance * size		0.164	0.263	0.623	0.533	-0.368	0.666
RetinaNet									
	Vanishing	distance	*	-5.130	0.374	-13.709	0.000	-5.886	-4.420
		size	*	1.912	0.249	7.695	0.000	1.440	2.415
		distance * size		0.152	0.588	0.259	0.795	-1.040	1.266
	Mislabeling	distance	*	-4.411	0.525	-8.405	0.000	-5.494	-3.437
		size	*	0.920	0.248	3.703	0.000	0.435	1.410
		distance * size		0.693	0.759	0.913	0.361	-0.872	2.103
	Untargeted	distance	*	-1.555	0.151	-10.285	0.000	-1.862	-1.270
		size	*	1.377	0.174	7.927	0.000	1.039	1.720
		distance * size	*	1.683	0.230	7.315	0.000	1.240	2.143
Faster R-CNN									
	Vanishing	distance	*	-6.113	0.604	-10.114	0.000	-7.351	-4.982
		size	*	1.638	0.275	5.964	0.000	1.114	2.192
		distance * size		-0.610	0.997	-0.611	0.541	-2.674	1.236
	Mislabeling	distance	*	-5.569	0.630	-8.839	0.000	-6.870	-4.398
		size		0.238	0.275	0.867	0.386	-0.301	0.780
		distance * size	*	2.237	0.767	2.918	0.004	0.578	3.597
	Untargeted	distance	*	-1.925	0.168	-11.451	0.000	-2.267	-1.607
		size	*	1.880	0.194	9.677	0.000	1.504	2.265
		distance * size	*	2.143	0.259	8.262	0.000	1.648	2.665
Cascade R-CNN									
	Vanishing	distance	*	-6.971	0.573	-12.163	0.000	-8.137	-5.890
		size	*	2.167	0.282	7.693	0.000	1.635	2.741
		distance * size		-0.329	0.883	-0.372	0.710	-2.161	1.309
	Mislabeling	distance	*	-6.440	0.585	-10.999	0.000	-7.639	-5.343
		size	*	0.486	0.237	2.049	0.040	0.023	0.955
		distance * size	*	1.829	0.798	2.292	0.022	0.144	3.280
	Untargeted	distance	*	-2.677	0.224	-11.971	0.000	-3.132	-2.255
		size	*	0.693	0.174	3.991	0.000	0.352	1.034
		distance * size	*	2.181	0.258	8.442	0.000	1.678	2.693

```
reg_mod <- reg_res$mod

newdata <- expand_grid(
  bbox_dist = linear_space(data$bbox_dist),
  bbox_size_perturb = linear_space(data$bbox_size_perturb)
) |>
  glimpse()
```

```
## Rows: 10,000
## Columns: 2
```

```
## $ bbox_dist          <dbl> 2e-04, 2e-04, 2e-04, 2e-04, 2e-04, 2e-04, 2e-04, 2e-~
## $ bbox_size_perturb <dbl> 0.000047817, 0.026654265, 0.053260713, 0.079867160, ~
```

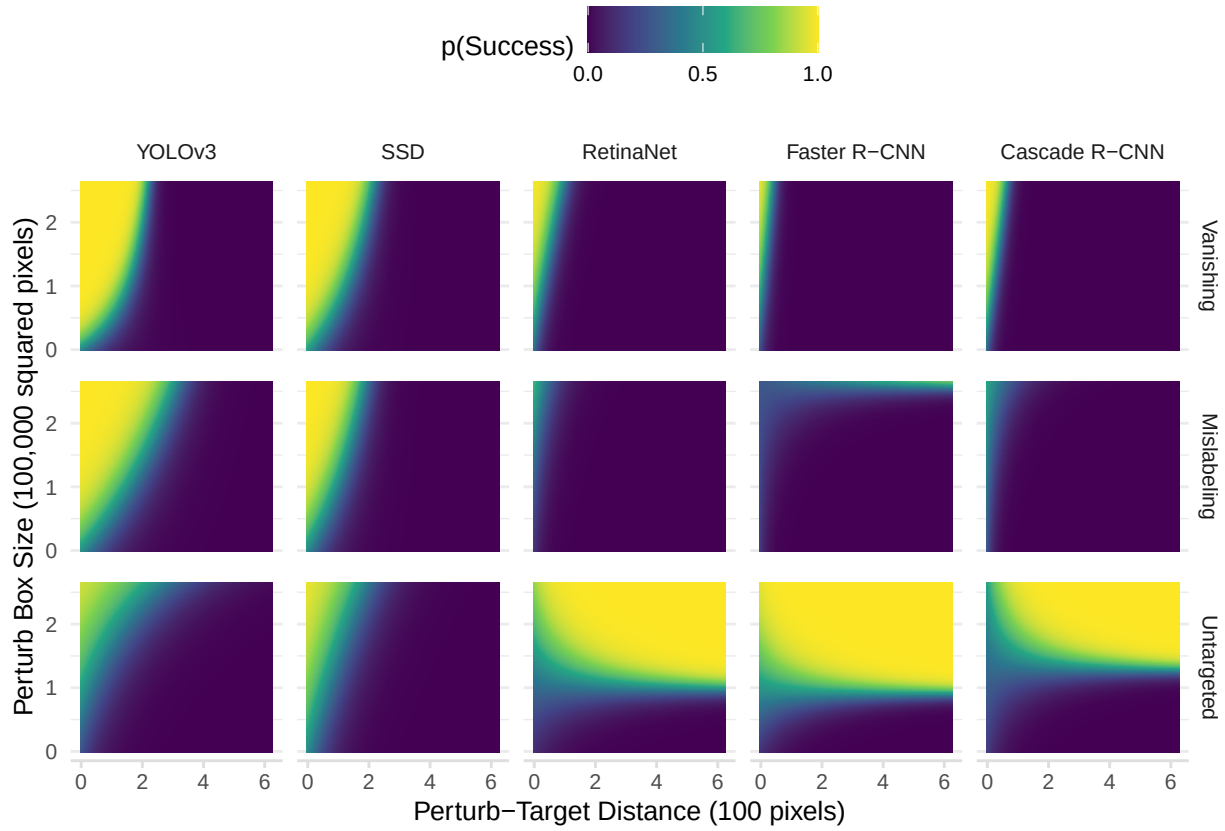
```
reg_pred <- reg_mod |>
  summarize(augment(mod, newdata = newdata, type.predict = "response")) |>
  rename(success = .fitted) |>
  glimpse()
```

```
## `summarise()` has grouped output by 'model_name', 'loss_target'. You can
## override using the `.groups` argument.
```

```
## Rows: 150,000
## Columns: 5
## Groups: model_name, loss_target [15]
## $ model_name         <ord> YOLOv3, YOLOv3, YOLOv3, YOLOv3, YOLOv3, YOLOv3, YOLO~
## $ loss_target        <ord> Vanishing, Vanishing, Vanishing, Vanishing, Vanishin~
## $ bbox_dist          <dbl> 2e-04, 2e-04, 2e-04, 2e-04, 2e-04, 2e-04, 2e-04, 2e-~
## $ bbox_size_perturb <dbl> 0.000047817, 0.026654265, 0.053260713, 0.079867160, ~
## $ success            <dbl> 0.4608801, 0.5036071, 0.5462815, 0.5882864, 0.629043~
```

```
g <- reg_pred |> ggplot(aes(bbox_dist, bbox_size_perturb, fill = success)) +
  geom_raster(interpolate = TRUE)

graph_dist_size(g)
```




```
# get success rate on ground truth sampled images
gt_success_data <- bbox_raw_data |>
  filter(bbox_type == "ground_truth") |>
  # loss_target is not relevant
```

```
  count(model_name, bbox_class, bbox_res_eval) |>
  # get success probability
  # https://stackoverflow.com/a/37448040/19655086
  group_by(model_name, bbox_class) |>
  summarise(gt_p_success = n[bbox_res_eval == "tp"] / sum(n)) |>
  glimpse()
```

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

```
## Rows: 399
## Columns: 3
## Groups: model_name [5]
## $ model_name   <ord> YOLOv3, YOLOv3, YOLOv3, YOLOv3, YOLOv3, YOLOv3, YOLOv3, Y~
## $ bbox_class   <chr> "airplane", "apple", "backpack", "banana", "baseball bat"~
## $ gt_p_success <dbl> 0.8866667, 0.6498516, 0.6024845, 0.6881533, 0.8106509, 0.~
```

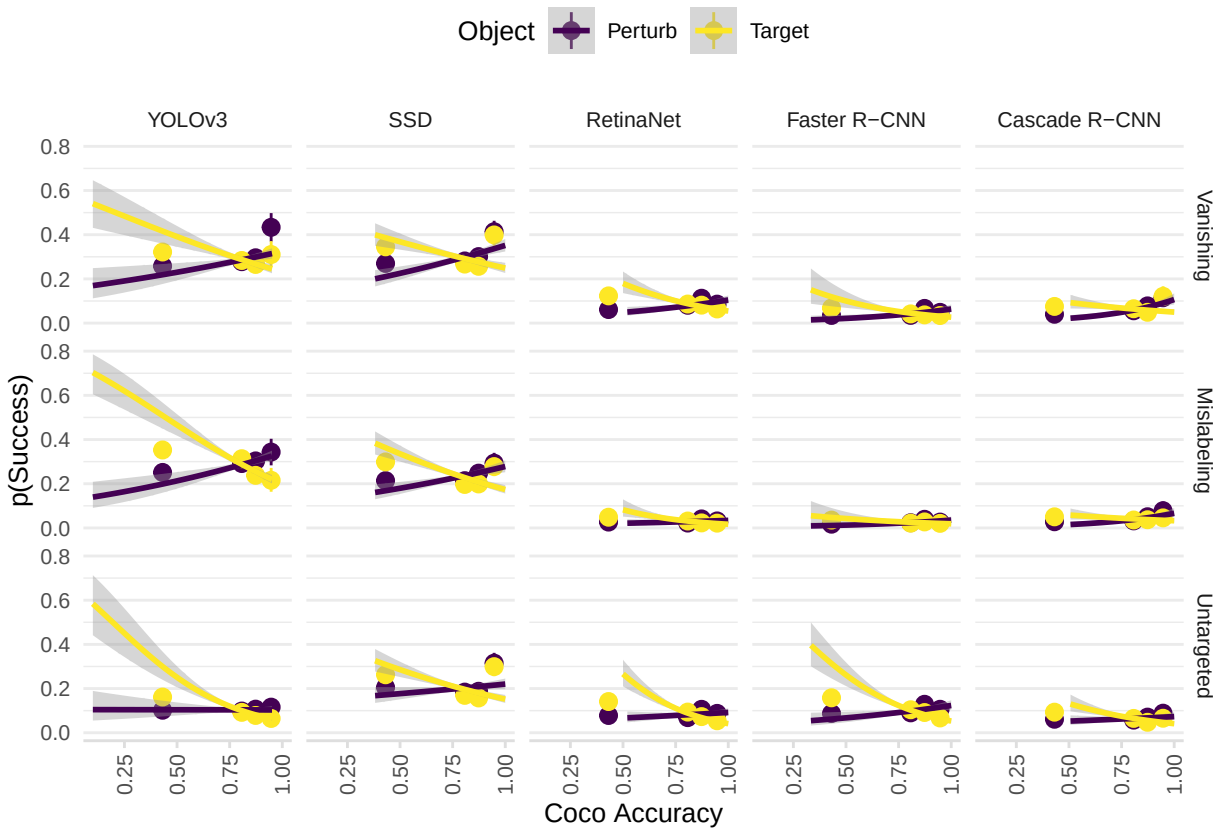
```
# by model_name, bbox_class
gt_success_data <- bbox_conf_data |>
  left_join(gt_success_data) |>
  glimpse()
```

```
## Joining, by = c("bbox_class", "model_name")
```

```
## Rows: 150,000
## Columns: 64
## $ fname                          <chr> "./data/random/results/random_repeat_10~
## $ sample_id                      <chr> "63baef4898dbbc7a545ee3a0", "63baef4898~
## $ sample_path                    <chr> "/projectsp/f_ps848_1/zhaobin/adversari~
## $ sample_width                   <int> 640, 425, 640, 416, 429, 385, 480, 478,~
## $ sample_height                  <int> 480, 640, 427, 640, 640, 308, 640, 640,~
## $ sample_mislabel_class_10       <chr> "truck", "train", "dog", "dog", "fire h~
## $ sample_mislabel_proba_10       <dbl> 1.266122e-03, 1.629095e-04, 3.693961e-0~
## $ sample_mislabel_class_50       <chr> "truck", "train", "dog", "dog", "fire h~
## $ sample_mislabel_proba_50       <dbl> 1.266122e-03, 1.629095e-04, 3.693961e-0~
## $ sample_mislabel_class_100      <chr> "truck", "train", "dog", "dog", "fire h~
## $ sample_mislabel_proba_100      <dbl> 1.266122e-03, 1.629095e-04, 3.693961e-0~
## $ sample_mislabel_class_200      <chr> "truck", "train", "dog", "dog", "fire h~
## $ sample_mislabel_proba_200      <dbl> 1.266122e-03, 1.629095e-04, 3.693961e-0~
## $ sample_mislabel_intended_200   <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_mislabel_50             <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_mislabel_intended_50    <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_mislabel_200            <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_vanish_10               <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_mislabel_intended_100   <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_mislabel_100            <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALS~
## $ sample_attack                  <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRU~
## $ bbox_id                        <chr> "63baef4e98dbbc7a545f1fe8", "63baef4e98~
## $ bbox_class                     <chr> "bus", "clock", "person", "teddy bear",~
## $ bbox_xywhn                     <list<double>> <0.05912231, 0.22379084, 0.761~
## $ bbox_conf                      <dbl> 0.9982775, 0.9760631, 0.8126031, 0.7020~
## $ bbox_res_eval                  <chr> "tp", "tp", "tp", "tp", "fp", "tp", "tp~
## $ bbox_iou_eval                  <dbl> 0.9772157, 0.7299863, 0.9619440, 0.7784~
## $ bbox_res_pgd_100_eval          <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_100_eval          <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
```

```
## $ bbox_res_pgd_100_mislabel_eval <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_100_mislabel_eval <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_pgd_10_eval           <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_10_eval           <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_pgd_10_mislabel_eval  <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_10_mislabel_eval  <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_pgd_200_eval          <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_200_eval          <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_pgd_200_mislabel_eval <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_200_mislabel_eval <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_pgd_50_eval           <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_50_eval           <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_pgd_50_mislabel_eval  <chr> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_iou_pgd_50_mislabel_eval  <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_res_predictions_eval      <chr> "tp", "tp", "tp", "tp", "tp", "tp", "tp~
## $ bbox_iou_predictions_eval      <dbl> 0.9772157, 0.7299863, 0.9619440, 0.7784~
## $ bbox_type                      <chr> "predictions", "predictions", "predicti~
## $ bbox_mislabel_50               <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_mislabel_100              <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_mislabel_200              <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ model_name                     <ord> Cascade R-CNN, Cascade R-CNN, Cascade R~
## $ loss_target                    <ord> Mislabeling, Mislabeling, Mislabeling, ~
## $ attack_bbox                    <chr> "ground_truth", "ground_truth", "ground~
## $ perturb_fun                    <chr> "perturb_inside", "perturb_inside", "pe~
## $ sample_vanish_50               <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ sample_vanish_200              <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ sample_mislabel_10             <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ sample_mislabel_intended_10    <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ sample_vanish_100              <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ bbox_mislabel_10               <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ target_or_perturb_boolean      <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRU~
## $ target_or_perturb              <ord> Target, Target, Target, Target, Target,~
## $ attack_itr                     <int> 200, 200, 200, 200, 200, 200, 200, 200,~
## $ success                        <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ~
## $ gt_p_success                   <dbl> 0.8574879, 0.8375000, 0.8941284, 0.8164~
```

```
gt_success_data |>
  graph_attr(gt_p_success, "COCO Accuracy")
```




```
pred_name <- "mean COCO accuracy for the target class"
main_pt <- "the results are mixed after controlling for target class confidence"

cap <- graph_caption(pred_name, glue("Although higher {pred_name} seem to decrease success rates, {main_
```

```
gt_success_graph <- gt_success_data |> filter(target_or_perturb == "Target")
gt_success_graph |>
  graph_attr(gt_p_success, pred_name)
```

```
model <- partial(glm_model, predictor = "gt_p_success * bbox_conf")
data <- gt_success_graph

reg_est <- get_tidied_reg(model, data)
```

```
## `summarise()` has grouped output by 'model_name', 'loss_target'. You can
## override using the `.groups` argument.
```

```
# there are both significantly positive and negative gt_p_success,
# and the interaction term is relatively large
ext_sig(reg_est, "neg", "gt_p_success")
```

```
## ----------gt_p_success----------
## Total 15 predictors:
## 9 (60%) significant;
## 5 (33%) neg
```

```
## # A tibble: 5 x 9
## # Groups:   model_name, loss_target [5]
##   model_name loss_target term    estim~1 std.e~2 stati~3 p.value conf.~4 conf.~5
```

```
##   <ord>      <ord>       <chr>     <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 YOLOv3     Untargeted  gt_p_s~   -3.28   1.33    -2.47   0.014   -5.91  -0.689
## 2 SSD        Untargeted  gt_p_s~   -2.09   0.867   -2.41   0.016   -3.79  -0.389
## 3 RetinaNet  Vanishing   gt_p_s~   -3.52   1.63    -2.16   0.031   -6.7   -0.309
## 4 RetinaNet  Mislabeling gt_p_s~   -7.90   2.93    -2.70   0.007  -13.6   -2.13
## 5 RetinaNet  Untargeted  gt_p_s~   -4.35   1.71    -2.55   0.011   -7.70  -1.01
## # ... with abbreviated variable names 1: estimate, 2: std.error, 3: statistic,
## #   4: conf.low, 5: conf.high
```

```
ext_sig(reg_est, "pos", "gt_p_success")
```

```
## ----------gt_p_success----------
## Total 15 predictors:
## 9 (60%) significant;
## 4 (27%) pos
```

```
## # A tibble: 4 x 9
## # Groups:   model_name, loss_target [4]
##   model_name    loss_tar~1 term  estim~2 std.e~3 stati~4 p.value conf.~5 conf.~6
##   <ord>         <ord>      <chr>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 Faster R-CNN  Vanishing  gt_p~    4.93    2.13    2.32   0.021   0.839    9.20
## 2 Faster R-CNN  Mislabeli~ gt_p~    8.52    2.77    3.08   0.002   3.22    14.1
## 3 Cascade R-CNN Mislabeli~ gt_p~    4.76    2.35    2.03   0.042   0.236    9.45
## 4 Cascade R-CNN Untargeted gt_p~    3.75    1.80    2.08   0.038   0.241    7.32
## # ... with abbreviated variable names 1: loss_target, 2: estimate,
## #   3: std.error, 4: statistic, 5: conf.low, 6: conf.high
```

```
ext_sig(reg_est, "both", "gt_p_success:bbox_conf")
```

```
## ----------gt_p_success:bbox_conf----------
## Total 15 predictors:
## 7 (47%) significant;
## 7 (47%) both
```

```
## # A tibble: 7 x 9
## # Groups:   model_name, loss_target [7]
##   model_name    loss_tar~1 term  estim~2 std.e~3 stati~4 p.value conf.~5 conf.~6
##   <ord>         <ord>      <chr>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 YOLOv3        Mislabeli~ gt_p~   -2.90    1.42   -2.05   0.041   -5.68  -0.118
## 2 RetinaNet     Mislabeli~ gt_p~   17.2     6.79    2.53   0.012    4.01  30.6
## 3 Faster R-CNN  Vanishing  gt_p~   -9.14    2.82   -3.24   0.001  -14.7   -3.63
## 4 Faster R-CNN  Mislabeli~ gt_p~  -12.7     3.89   -3.27   0.001  -20.3   -5.04
## 5 Faster R-CNN  Untargeted gt_p~   -5.15    1.97   -2.61   0.009   -9.00  -1.27
## 6 Cascade R-CNN Mislabeli~ gt_p~   -6.49    3.24   -2.00   0.045  -12.8   -0.119
## 7 Cascade R-CNN Untargeted gt_p~   -6.89    2.52   -2.73   0.006  -11.8   -1.93
## # ... with abbreviated variable names 1: loss_target, 2: estimate,
## #   3: std.error, 4: statistic, 5: conf.low, 6: conf.high
```

```
print_statistics(reg_est, table_caption(
  glue("{pred_name}, with target confidence as covariate,"),
  glue("{main_pt} and the relatively large interaction terms make interpretation challenging")
))
```

Table 6: We run a logistic model regressing success against mean COCO accuracy for the target class, with target confidence as covariate, in the randomized attack experiment. The results are mixed after controlling for target class confidence and the relatively large interaction terms make interpretation challenging. Table headers are explained in Appendix **??**.

Group	Regression							
Attack	term	sig	estimate	std.error	statistic	p.value	conf.low	conf.high
YOLOv3								
Vanishing	accuracy		-0.436	1.053	-0.414	0.679	-2.497	1.634
	confidence		0.475	1.145	0.415	0.678	-1.771	2.722
	accuracy * confidence		-1.233	1.414	-0.873	0.383	-4.007	1.538
Mislabeling	accuracy		0.159	1.034	0.154	0.877	-1.871	2.186
	confidence		0.538	1.147	0.469	0.639	-1.716	2.782
	accuracy * confidence	*	-2.902	1.418	-2.047	0.041	-5.679	-0.118
Untargeted	accuracy	*	-3.282	1.330	-2.468	0.014	-5.906	-0.689
	confidence	*	-4.205	1.659	-2.535	0.011	-7.517	-1.009
	accuracy * confidence		1.199	2.069	0.580	0.562	-2.799	5.314
SSD								
Vanishing	accuracy		-0.743	0.817	-0.909	0.363	-2.342	0.864
	confidence		-0.042	0.869	-0.048	0.962	-1.746	1.664
	accuracy * confidence		-0.396	1.089	-0.363	0.716	-2.532	1.740
Mislabeling	accuracy		-0.797	0.842	-0.946	0.344	-2.444	0.857
	confidence		-0.277	0.910	-0.305	0.761	-2.064	1.504
	accuracy * confidence		-0.977	1.145	-0.853	0.394	-3.221	1.271
Untargeted	accuracy	*	-2.087	0.867	-2.408	0.016	-3.789	-0.389
	confidence	*	-3.125	0.990	-3.157	0.002	-5.081	-1.198
	accuracy * confidence		1.486	1.241	1.198	0.231	-0.933	3.932
RetinaNet								
Vanishing	accuracy	*	-3.520	1.630	-2.160	0.031	-6.700	-0.309
	confidence	*	-6.644	2.646	-2.511	0.012	-11.879	-1.504
	accuracy * confidence		4.770	3.090	1.544	0.123	-1.259	10.858
Mislabeling	accuracy	*	-7.902	2.929	-2.697	0.007	-13.606	-2.128
	confidence	*	-20.491	5.908	-3.469	0.001	-32.313	-9.179
	accuracy * confidence	*	17.153	6.788	2.527	0.012	4.011	30.592
Untargeted	accuracy	*	-4.352	1.707	-2.549	0.011	-7.701	-1.007
	confidence	*	-9.046	2.985	-3.030	0.002	-14.984	-3.279
	accuracy * confidence		5.142	3.520	1.461	0.144	-1.699	12.099
Faster R-CNN								
Vanishing	accuracy	*	4.935	2.132	2.315	0.021	0.839	9.204
	confidence	*	4.666	2.288	2.040	0.041	0.200	9.180
	accuracy * confidence	*	-9.142	2.824	-3.238	0.001	-14.707	-3.627
Mislabeling	accuracy	*	8.515	2.767	3.078	0.002	3.222	14.084

	confidence	*	6.412	3.183	2.015	0.044	0.114	12.625
	accuracy * confidence	*	-12.713	3.888	-3.270	0.001	-20.304	-5.038
Untargeted	accuracy		1.447	1.442	1.003	0.316	-1.373	4.289
	confidence		0.733	1.588	0.462	0.644	-2.395	3.836
	accuracy * confidence	*	-5.146	1.969	-2.614	0.009	-8.999	-1.273
Cascade R-CNN								
Vanishing	accuracy		1.644	2.104	0.781	0.435	-2.419	5.840
	confidence		0.766	2.173	0.353	0.724	-3.466	5.064
	accuracy * confidence		-2.987	2.679	-1.115	0.265	-8.273	2.237
Mislabeling	accuracy	*	4.762	2.347	2.029	0.042	0.236	9.446
	confidence		1.915	2.651	0.722	0.470	-3.301	7.107
	accuracy * confidence	*	-6.491	3.243	-2.002	0.045	-12.843	-0.119
Untargeted	accuracy	*	3.752	1.805	2.079	0.038	0.241	7.324
	confidence		1.669	2.033	0.821	0.412	-2.339	5.637
	accuracy * confidence	*	-6.887	2.519	-2.734	0.006	-11.812	-1.930

```
# restrict to mislabeling and largest attack_itr
bbox_proba_graph <- bbox_conf_data |>
  filter(loss_target == "Mislabeling" & target_or_perturb == "Target") |>
  rename(sample_mislabel_proba = glue("sample_mislabel_proba_{max(bbox_conf_data$attack_itr)}"))

# check is not logit
stopifnot(max(bbox_proba_graph$sample_mislabel_proba) <= 1 && min(bbox_proba_graph$sample_mislabel_proba
```

```
pred_name <- "intended class probability"
att_name <- "for the mislabeling attack"

main_pt <- glue("does not predict success rates after controlling for target class confidence, except f
cap <- graph_caption(pred_name, glue("Although {pred_name} seem to increase success rates {att_name}, i

g <- bbox_proba_graph |>
  graph_attr(sample_mislabel_proba, glue("{pred_name} {att_name}"), scale_x_log10())
```

```
model <- partial(glm_model, predictor = "log(sample_mislabel_proba) * bbox_conf")
data <- bbox_proba_graph

reg_est <- get_tidied_reg(model, data)
```

```
## `summarise()` has grouped output by 'model_name', 'loss_target'. You can
## override using the `.groups` argument.
```

```
ext_sig(reg_est, "pos", "log(sample_mislabel_proba)")
```

```
## ----------log(sample_mislabel_proba)----------
## Total 5 predictors:
## 1 (20%) significant;
## 1 (20%) pos
```

```
## # A tibble: 1 x 9
## # Groups:   model_name, loss_target [1]
##   model_name loss_target term    estim~1 std.e~2 stati~3 p.value conf.~4 conf.~5
```

```
##   <ord>      <ord>       <chr>     <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 RetinaNet  Mislabeling log(sa~   0.626    0.31    2.02   0.044   0.003    1.22
## # ... with abbreviated variable names 1: estimate, 2: std.error, 3: statistic,
## #   4: conf.low, 5: conf.high
```

```
ext_sig(reg_est, "both", "log(sample_mislabel_proba):bbox_conf")
```

```
## ----------log(sample_mislabel_proba):bbox_conf----------
## Total 5 predictors:
## 0 (0%) significant;
## 0 (0%) both
```

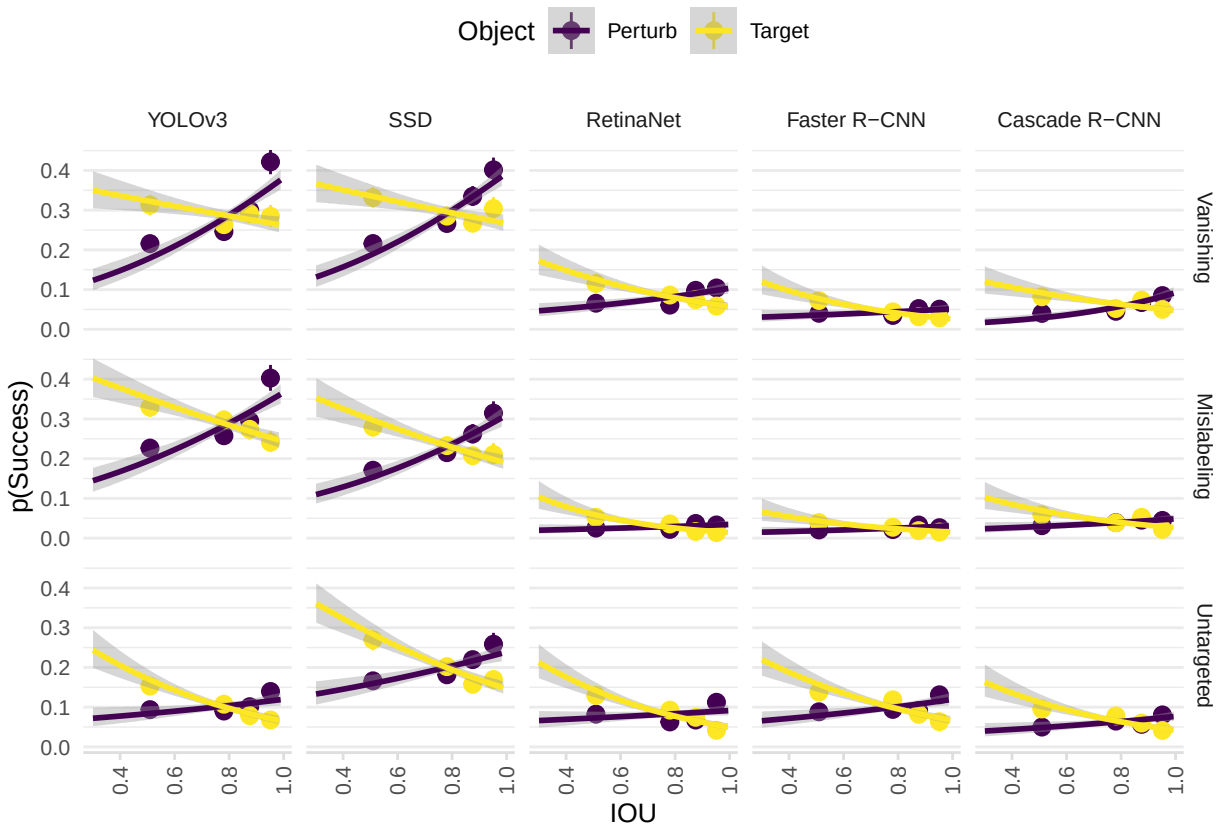
```
## # A tibble: 0 x 9
## # Groups:   model_name, loss_target [0]
## # ... with 9 variables: model_name <ord>, loss_target <ord>, term <chr>,
## #   estimate <dbl>, std.error <dbl>, statistic <dbl>, p.value <dbl>,
## #   conf.low <dbl>, conf.high <dbl>
```

```
print_statistics(reg_est, table_caption(glue("log({pred_name}) {att_name}, with predicted class's confi
```

Table 7: We run a logistic model regressing success against log(intended class probability) for the mislabeling attack, with predicted class's confidence as covariate, in the randomized attack experiment. Intended class probability does not predict success rates after controlling for target class confidence, except for RetinaNet. Table headers are explained in Appendix **??**.

Group		Regression						
Model	term	sig	estimate	std.error	statistic	p.value	conf.low	conf.high
Mislabeling								
YOLOv3	log(probability)		0.052	0.036	1.425	0.154	-0.019	0.123
	confidence	*	-1.777	0.431	-4.125	0.000	-2.624	-0.935
	log(probability) * confidence		0.010	0.050	0.193	0.847	-0.089	0.108
SSD	log(probability)		0.034	0.042	0.791	0.429	-0.049	0.117
	confidence	*	-0.760	0.375	-2.026	0.043	-1.494	-0.023
	log(probability) * confidence		0.025	0.054	0.473	0.636	-0.080	0.131
RetinaNet	log(probability)	*	0.626	0.310	2.018	0.044	0.003	1.218
	confidence	*	-8.462	1.804	-4.691	0.000	-12.016	-4.950
	log(probability) * confidence		-1.016	0.648	-1.568	0.117	-2.242	0.295
Faster R-CNN	log(probability)		0.060	0.100	0.593	0.553	-0.137	0.257
	confidence	*	-3.022	0.925	-3.265	0.001	-4.835	-1.201
	log(probability) * confidence		0.076	0.147	0.521	0.603	-0.207	0.368
Cascade R-CNN	log(probability)		0.117	0.080	1.474	0.140	-0.039	0.274
	confidence	*	-2.872	0.722	-3.979	0.000	-4.283	-1.450
	log(probability) * confidence		-0.014	0.108	-0.130	0.897	-0.224	0.200

```
# bbox iou always based on predictions bbox like confidence
bbox_conf_data |>
  graph_attr(bbox_iou_predictions_eval, " IOU ")
```




```
# restrict to target bbox and untargeted attack only
pred_name <- "target iou for the untargeted attack"
main_pt <- glue("{pred_name} increases success rates on all models")

cap <- graph_caption(pred_name, main_pt)

bbox_iou_graph <- bbox_conf_data |> filter(target_or_perturb == "Target" & loss_target == "Untargeted")
bbox_iou_graph |>
  graph_attr(bbox_iou_predictions_eval, pred_name)
```

```
model <- partial(glm_model, predictor = "bbox_iou_predictions_eval")
data <- bbox_iou_graph

reg_est <- get_tidied_reg(model, data)
```

```
## `summarise()` has grouped output by 'model_name', 'loss_target'. You can
## override using the `.groups` argument.
```

```
ext_sig(reg_est, "neg")
```

```
## Total 5 predictors:
## 5 (100%) significant;
## 5 (100%) neg
```

```
## # A tibble: 5 x 9
## # Groups:   model_name, loss_target [5]
##   model_name    loss_tar~1 term  estim~2 std.e~3 stati~4 p.value conf.~5 conf.~6
##   <ord>         <ord>      <chr>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
## 1 YOLOv3        Untargeted bbox~   -2.20   0.278   -7.90       0   -2.74   -1.65
```

```
## 2 SSD           Untargeted bbox~   -1.70   0.226   -7.55       0   -2.15   -1.26
## 3 RetinaNet     Untargeted bbox~   -2.34   0.275   -8.53       0   -2.88   -1.80
## 4 Faster R-CNN  Untargeted bbox~   -1.96   0.267   -7.34       0   -2.48   -1.43
## 5 Cascade R-CNN Untargeted bbox~   -2.12   0.307   -6.90       0   -2.72   -1.51
## # ... with abbreviated variable names 1: loss_target, 2: estimate,
## #   3: std.error, 4: statistic, 5: conf.low, 6: conf.high
```

```
print_statistics(reg_est, table_caption(pred_name, main_pt))
```

Table 8: We run a logistic model regressing success against target IOU for the untargeted attack in the randomized attack experiment. Target IOU for the untargeted attack increases success rates on all models. Table headers are explained in Appendix **??**.

Group	Regression							
Model	term	sig	estimate	std.error	statistic	p.value	conf.low	conf.high
Untargeted								
YOLOv3	iou	*	-2.199	0.278	-7.904	0	-2.741	-1.650
SSD	iou	*	-1.704	0.226	-7.549	0	-2.146	-1.260
RetinaNet	iou	*	-2.344	0.275	-8.533	0	-2.880	-1.802
Faster R-CNN	iou	*	-1.961	0.267	-7.340	0	-2.481	-1.433
Cascade R-CNN	iou	*	-2.122	0.307	-6.902	0	-2.718	-1.512

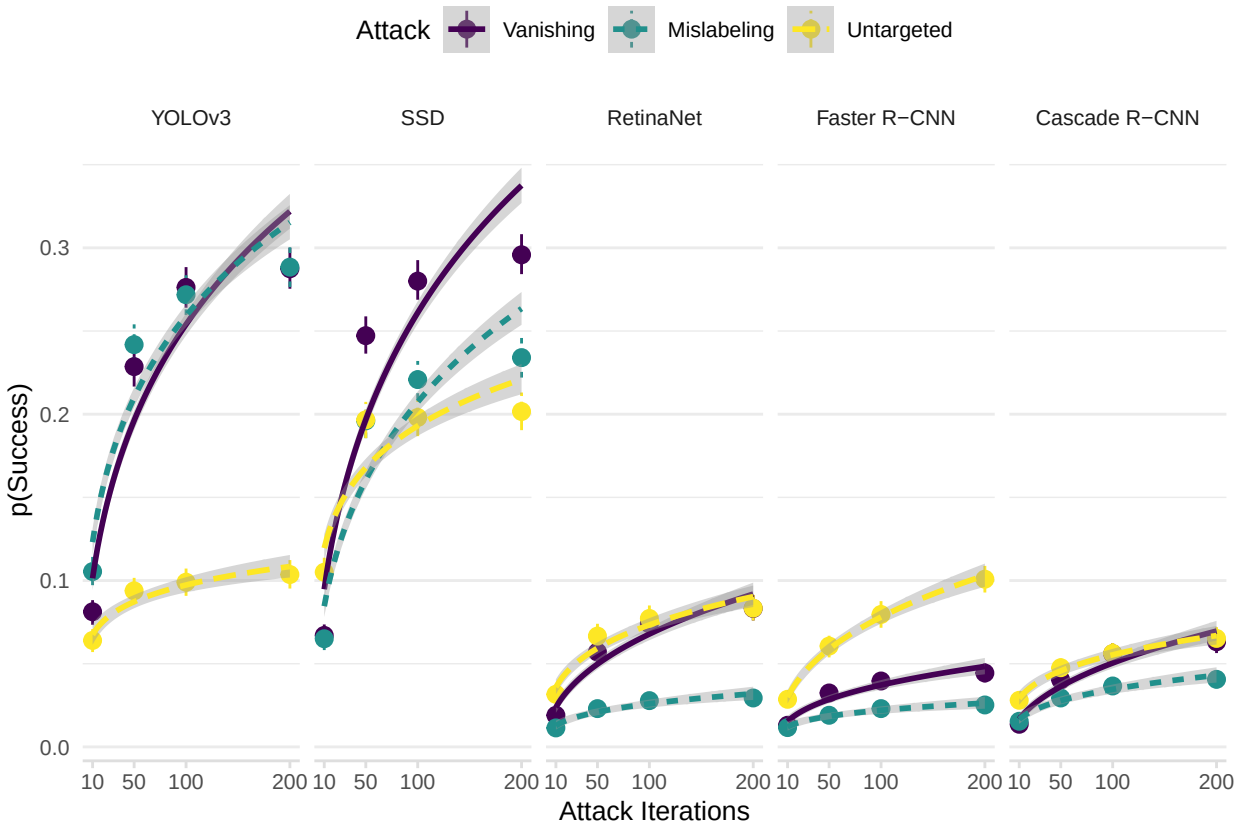


Figure 1: **Intent obfuscating attack is feasible for all models and attacks:** We conduct a randomized experiment by resampling COCO images, and within those images randomly sampling correctly predicted target and perturb objects. Then we distort the perturb objects to disrupt the target objects varying the attack iterations. The binned summaries and regression trendlines graph success proportion against attack iterations in the randomized attack experiment. Errors are 95% confidence intervals. and every point aggregates success over 5,000 images. Targeted vanishing and mislabeling attacks obtain significantly greater success on the 1-stage YOLOv3 and SSD than the 2-stage Faster R-CNN and Cascade R-CNN detectors. However, the 1-stage RetinaNet is as resilient as the 2-stage detectors. Additionally, targeted attacks are significantly more successful than untargeted attacks on YOLOv3 and SSD, but the pattern does not exist for RetinaNet, Faster R-CNN, and Cascade R-CNN. Within targeted attacks, vanishing achieves significantly greater success than mislabeling attack on all models except YOLOv3. Moreover, success rates significantly increase with larger attack iterations. Significance is determined at $\alpha < 0.05$ using a Wald z-test on the logistic estimates. Full details are given in Section **??**.

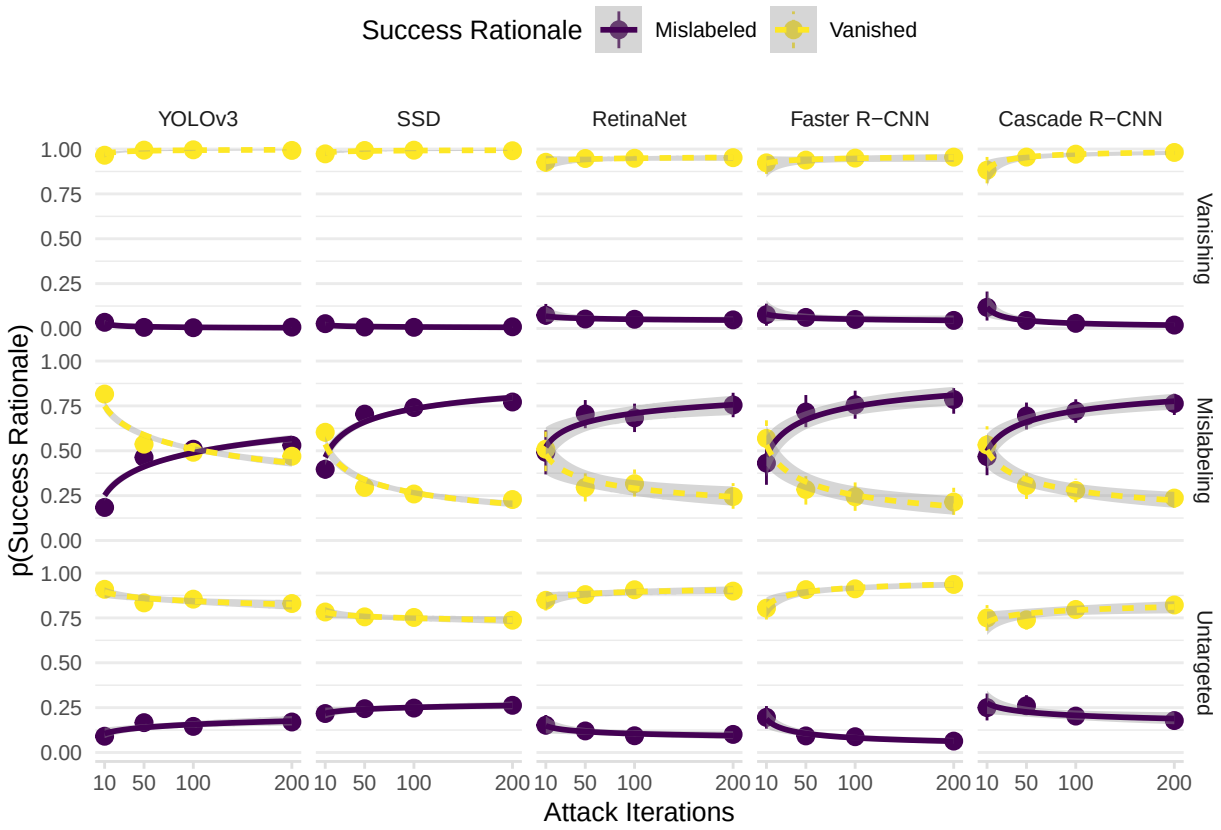


Figure 2: **Vanishing and mislabeling attacks mostly cause target objects to vanish and get mislabeled:** The graph breaks down the success rationale within the success cases (Figure 1). Though we did not restrict success to the intended attack mode (e.g. a vanishing attack which mislabels the target object still count as a success case), the target objects do vanish and get mislabeled in most success cases respectively in the vanishing and mislabeling attacks. The binned summaries and regression trendlines break down the success cases into proportion vanished and mislabeled—separated by attack—against attack iterations in the randomized attack experiment. Errors are 95% confidence intervals.

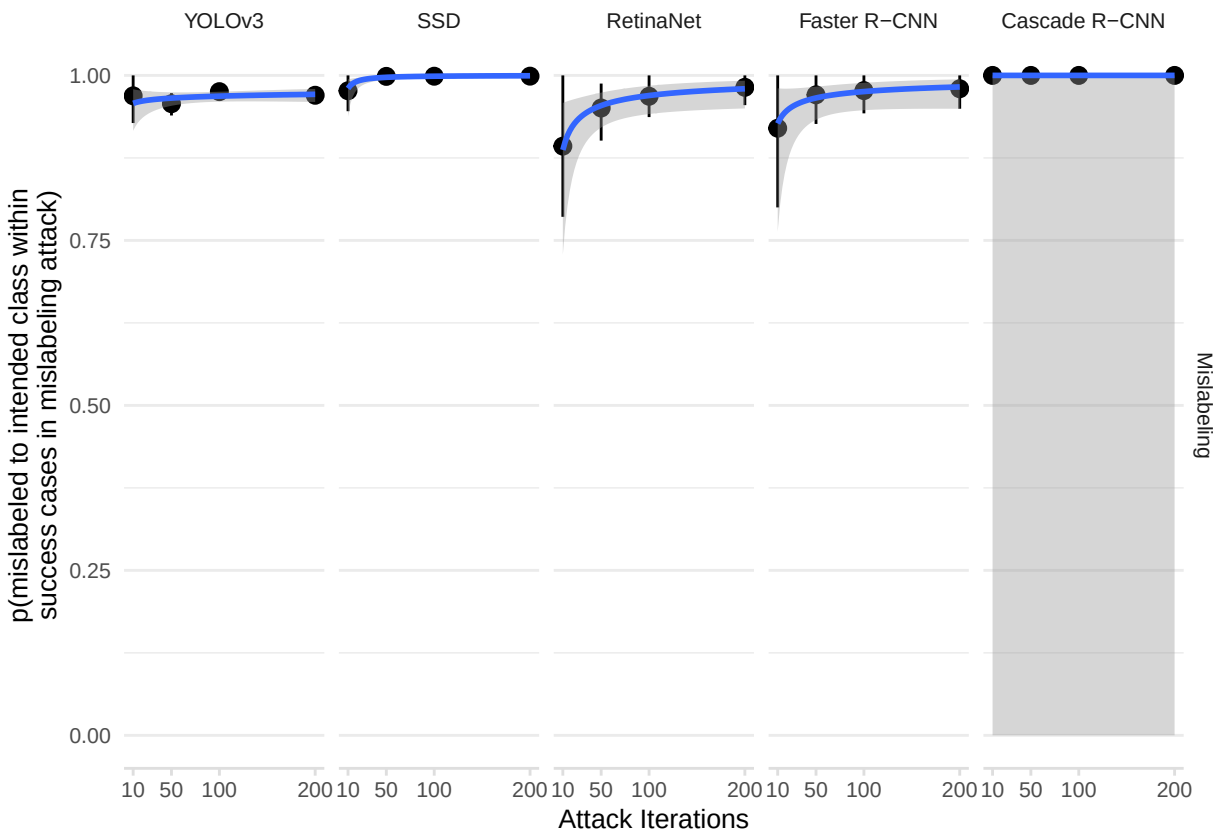


Figure 3: **Mislabeling attacks usually mislabel the target objects to the intended class:** The binned summaries and regression trendlines give us the proportion mislabeled to the intended class within the success cases in the mislabeling attack. The proportion is plotted against attack iterations in the randomized attack experiment. Errors are 95% confidence intervals. For Cascade R-CNN, the logistic model did not converge because the mislabel intended proportion is constant at 100%.

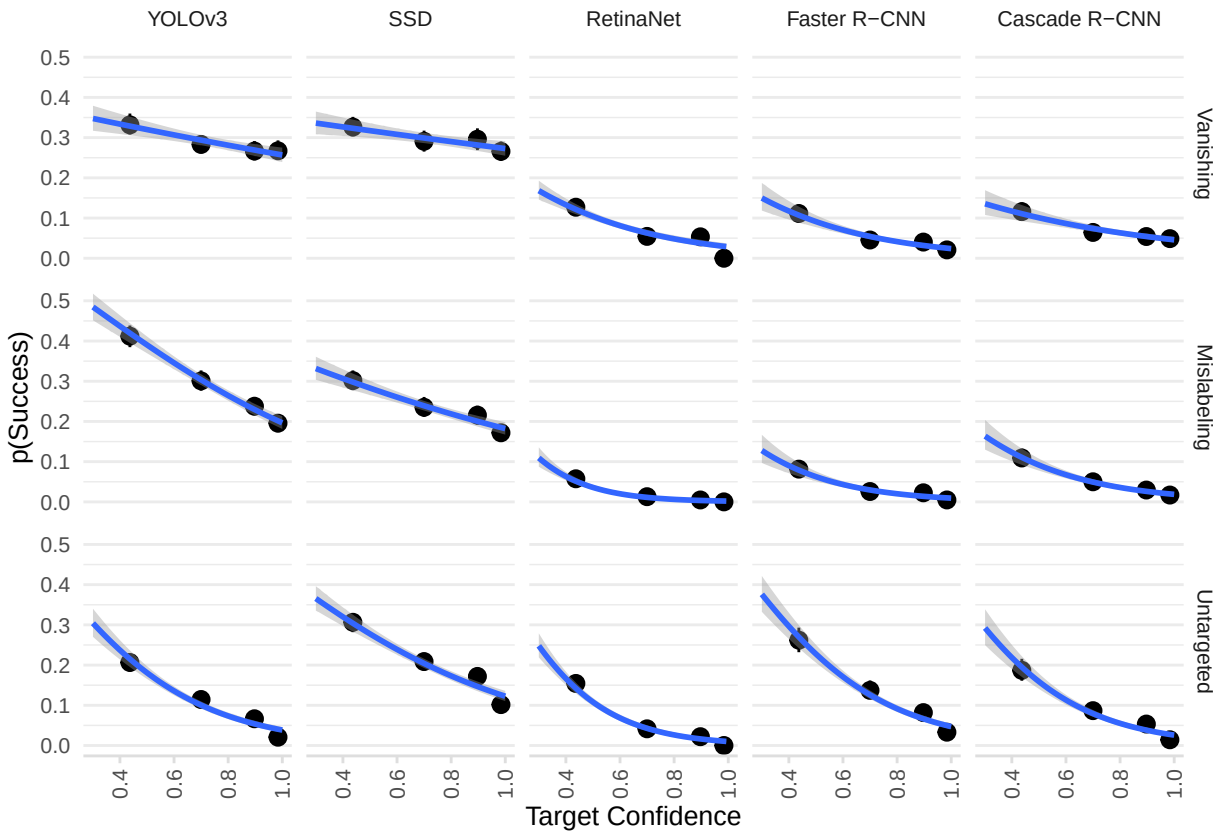


Figure 4: **Lower target confidence significantly increases success rates for all models and attacks:** The binned summaries and regression trendlines graph success proportion against target confidence in the randomized attack experiment. Bins are split into quantiles. Errors are 95% confidence intervals.

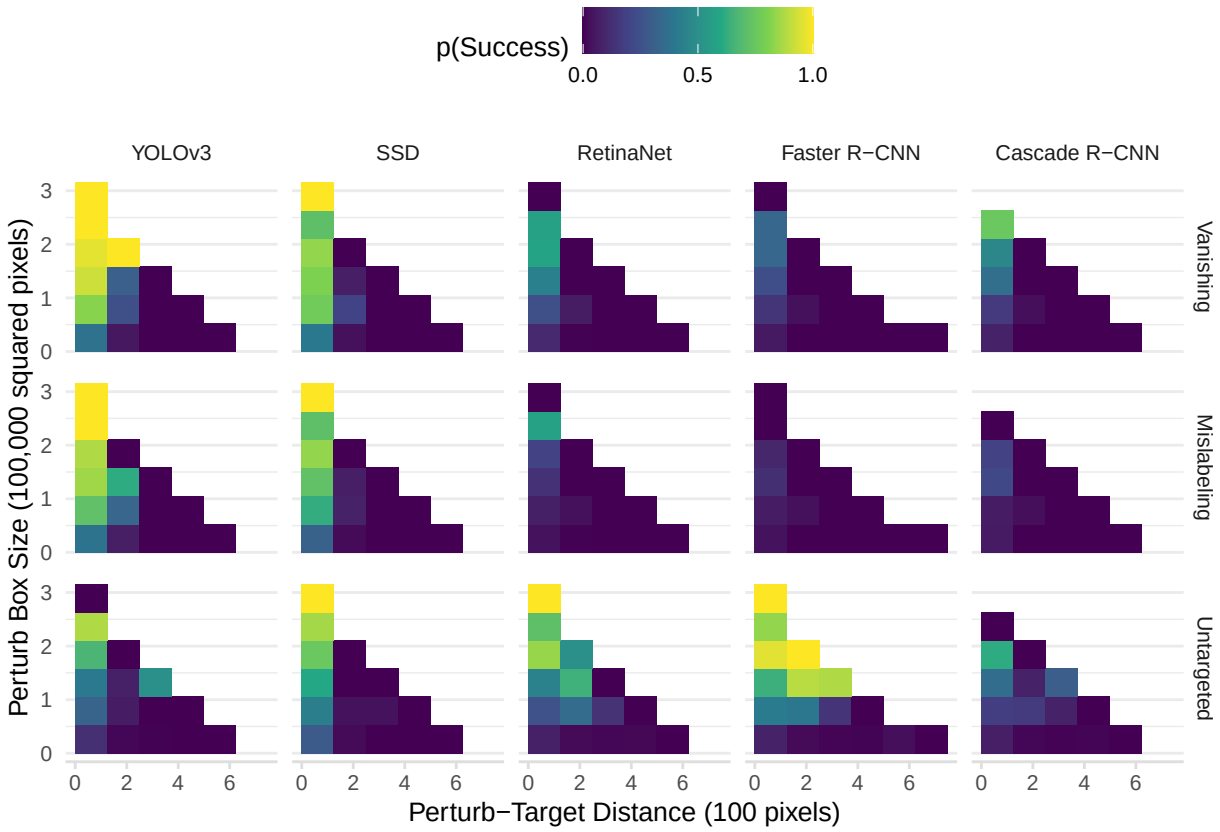


Figure 5: **Larger perturb objects significantly increase success rates for all models and attacks, except for mislabeling attack on Faster R-CNN, after controlling for perturb-target distances; Shorter perturb-target distances significantly increase success rates for all models and attacks, after controlling for perturb object sizes:** The binned summaries graph success proportion against perturb-target distance (100 pixels) and perturb box size (100,000 squared pixels) in the randomized attack experiment.

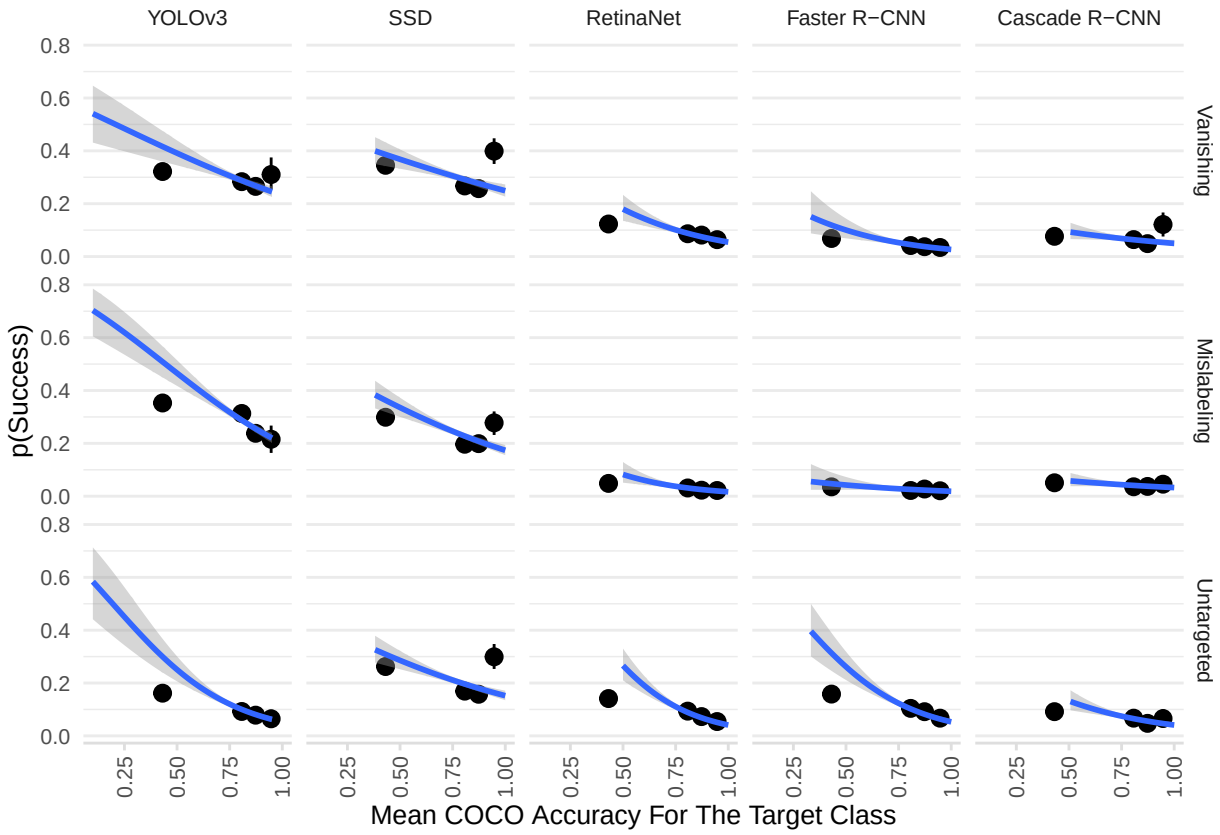


Figure 6: **Although higher mean COCO accuracy for the target class seem to decrease success rates, the results are mixed after controlling for target class confidence (Table 6):** The binned summaries and regression trendlines graph success proportion against mean COCO accuracy for the target class in the randomized attack experiment. Bins are split into quantiles. Errors are 95% confidence intervals.

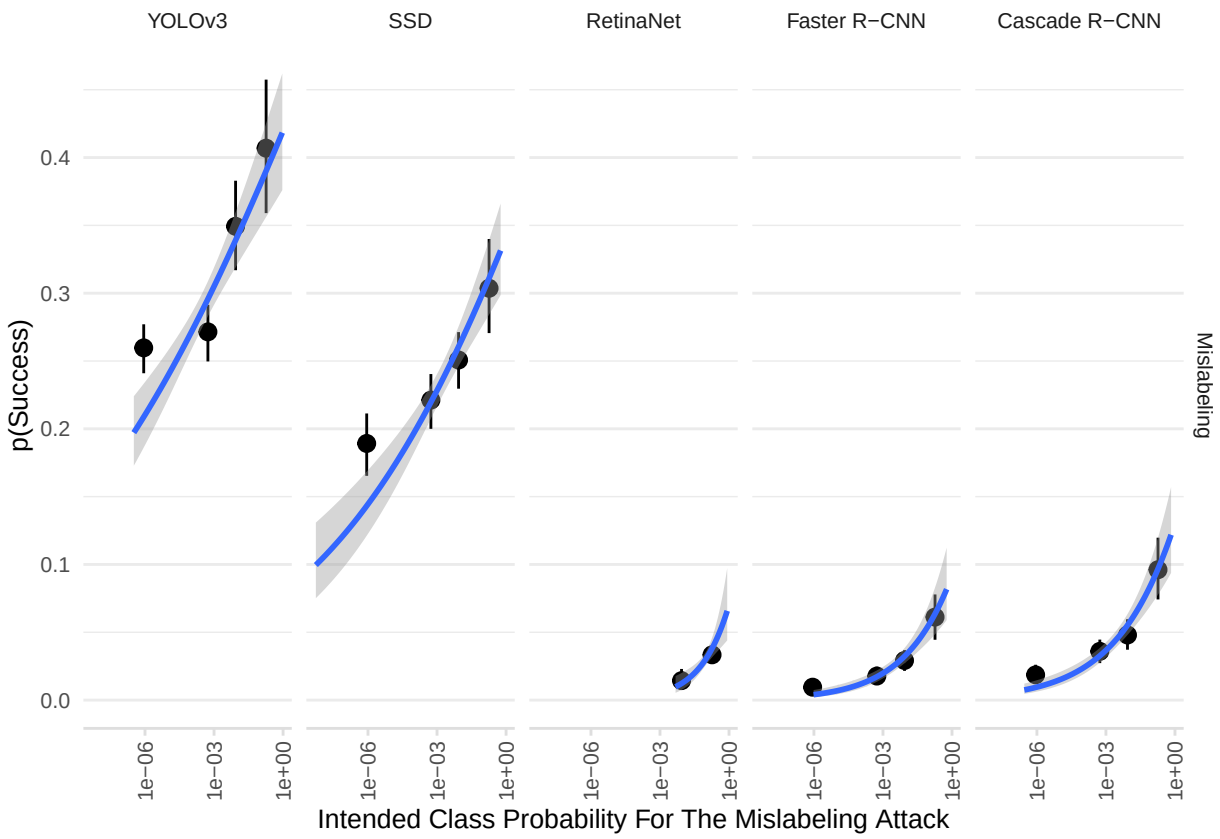


Figure 7: **Although intended class probability seem to increase success rates for the mislabeling attack, it does not predict success rates after controlling for target class confidence, except for RetinaNet (Table 7):** The binned summaries and regression trendlines graph success proportion against intended class probability in the randomized attack experiment. Bins are split into quantiles. Errors are 95% confidence intervals.

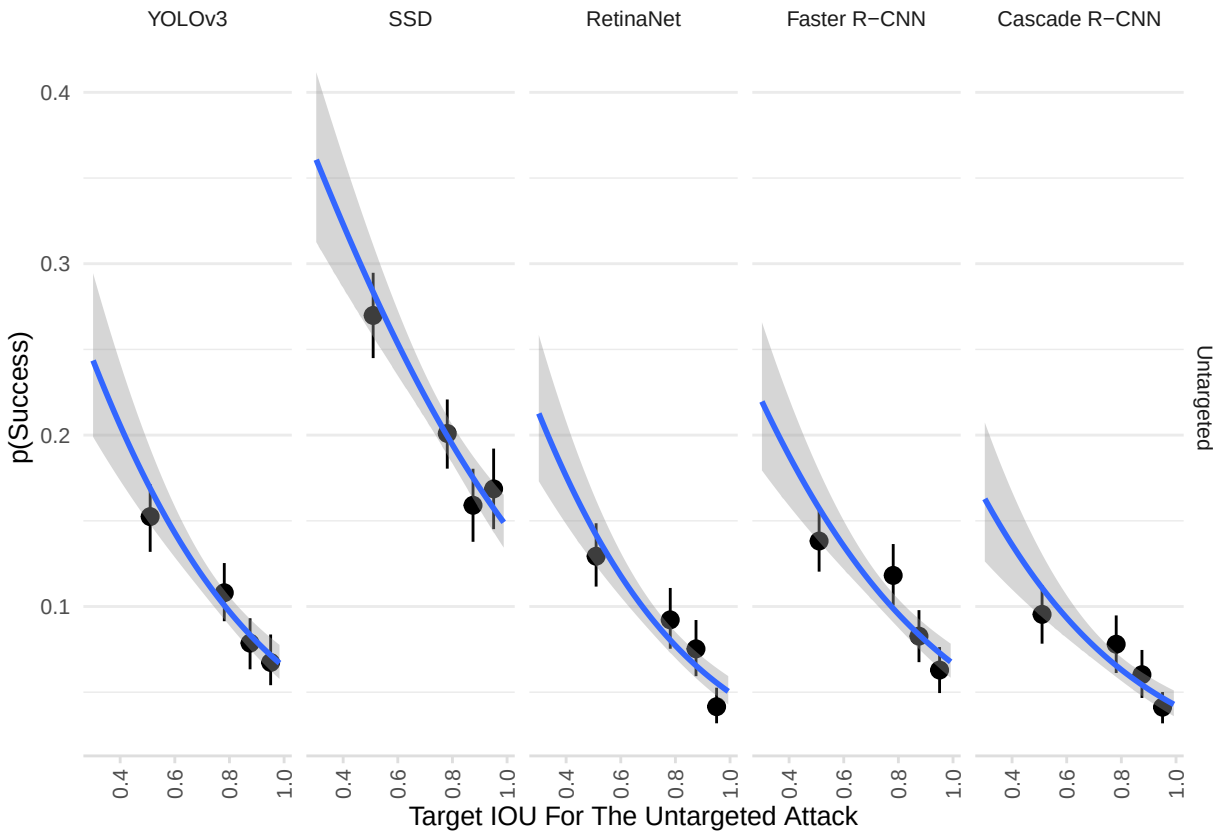


Figure 8: **Target IOU for the untargeted attack increases success rates on all models:** The binned summaries and regression trendlines graph success proportion against target IOU for the untargeted attack in the randomized attack experiment. Bins are split into quantiles. Errors are 95% confidence intervals.