alarm.R

garyw

2021-04-30

```r
# # Install packages
# install.packages(c("bnlearn", "bnviewer"))
# if (!requireNamespace("BiocManager", quietly = TRUE))
#   install.packages("BiocManager")
# BiocManager::install()
# BiocManager::install(c("graph", "Rgraphviz"))
# install.packages("ggplot2")

# Load packages
library("bnlearn")
library("bnviewer")
library("Rgraphviz")
```

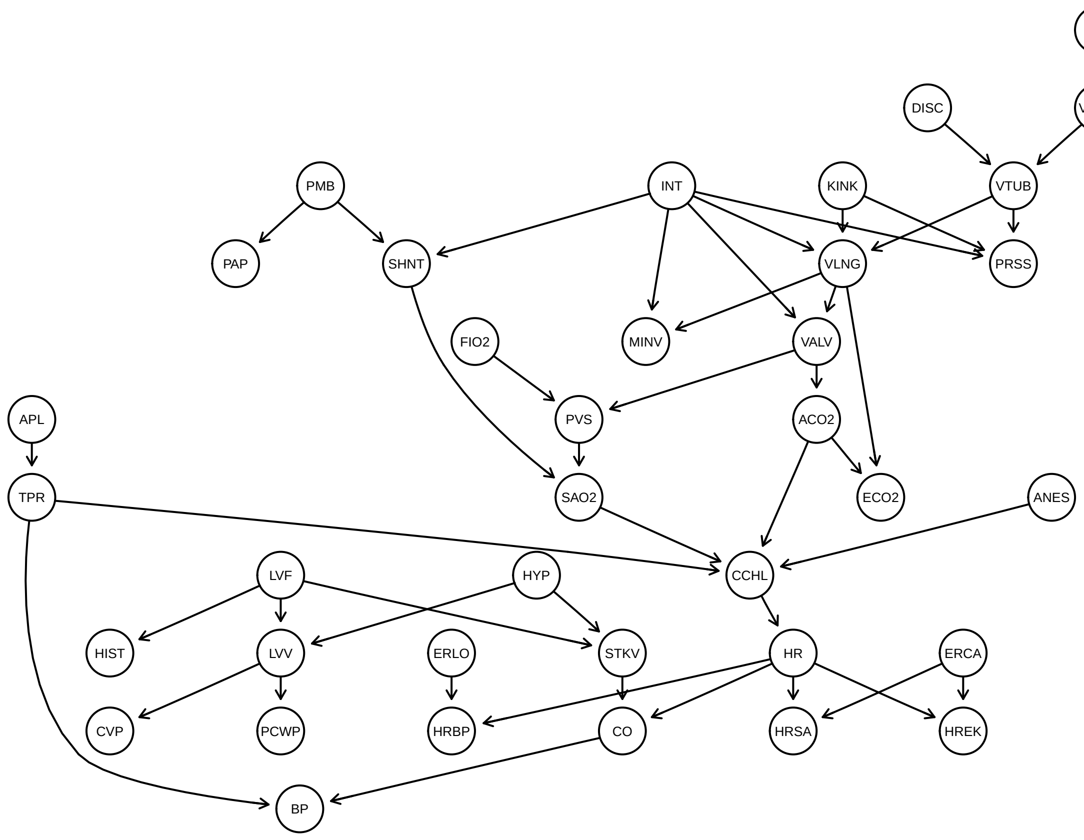
```
## Loading required package: graph

## Loading required package: BiocGenerics

## Loading required package: parallel

##
## Attaching package: 'BiocGenerics'

## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB

## The following objects are masked from 'package:bnlearn':
##
##     path, score

## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs

## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, basename, cbind, colnames,
##     dirname, do.call, duplicated, eval, evalq, Filter, Find, get, grep,
##     grepl, intersect, is.unsorted, lapply, Map, mapply, match, mget,
##     order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
##     rbind, Reduce, rownames, sapply, setdiff, sort, table, tapply,
##     union, unique, unsplit, which.max, which.min

##
## Attaching package: 'graph'
## The following objects are masked from 'package:bnlearn':
##
##     degree, nodes, nodes<-
```

```
## Loading required package: grid
```

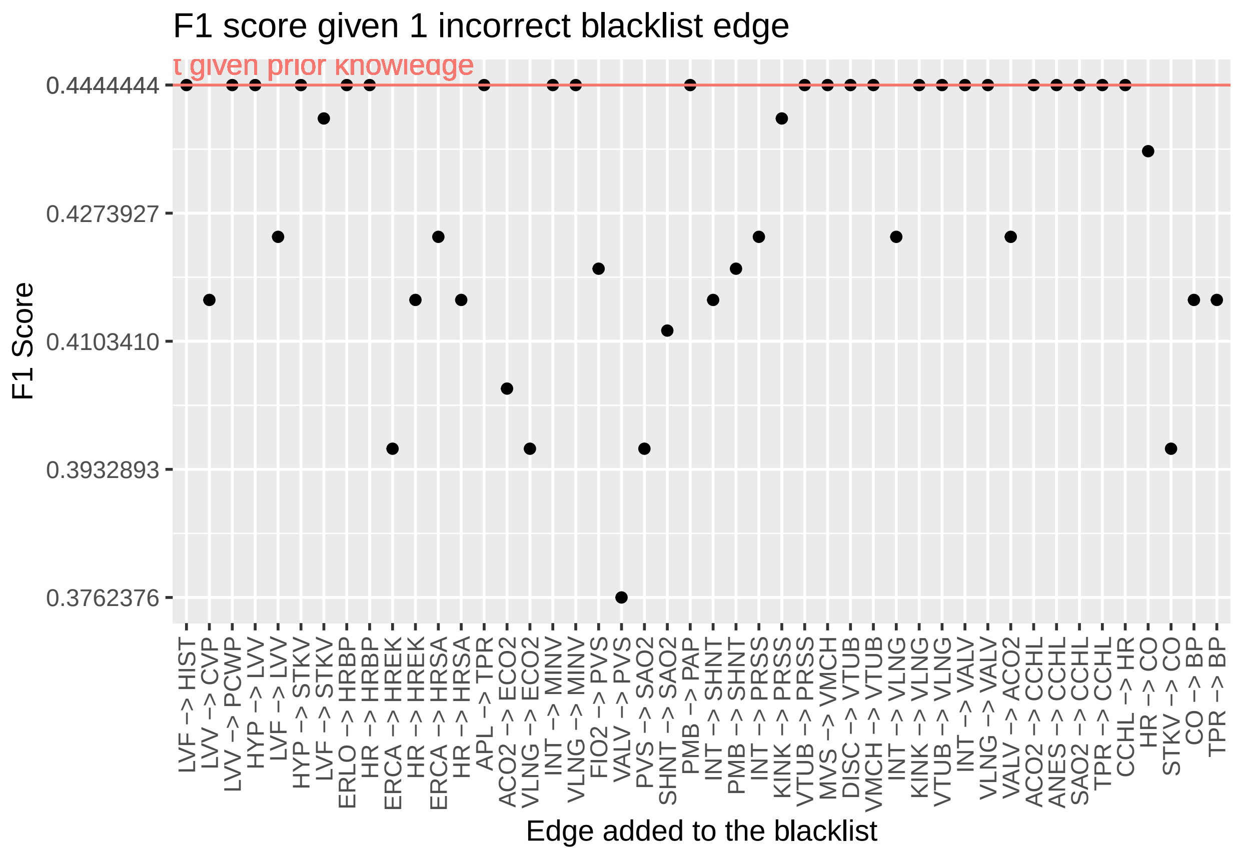
```
library("ggplot2")

# setwd('~/Projects/cics490e_research')
setwd('E:/Projects/cics490e_research')

# Compute f1 score given tp, fp, fn
f1 <- function(m) {
  tp <- m$tp
  fp <- m$fp
  fn <- m$fn

  return(tp / (tp + (fp + fn) / 2))
}

# Load Dataset
data('alarm')
head(alarm)
```

```
##      CVP   PCWP  HIST    TPR     BP     CO HRBP HREK HRSA    PAP   SAO2   FIO2
## 1 NORMAL NORMAL FALSE    LOW NORMAL   HIGH HIGH HIGH HIGH NORMAL NORMAL    LOW
## 2 NORMAL NORMAL FALSE NORMAL    LOW    LOW HIGH HIGH HIGH NORMAL    LOW NORMAL
## 3 NORMAL   HIGH FALSE NORMAL NORMAL   HIGH HIGH HIGH HIGH NORMAL    LOW NORMAL
## 4 NORMAL NORMAL FALSE    LOW    LOW   HIGH HIGH HIGH HIGH NORMAL NORMAL NORMAL
## 5 NORMAL NORMAL FALSE    LOW    LOW NORMAL HIGH HIGH HIGH NORMAL    LOW NORMAL
## 6 NORMAL NORMAL FALSE    LOW NORMAL   HIGH HIGH HIGH HIGH NORMAL    LOW NORMAL
##     PRSS ECO2 MINV    MVS   HYP   LVF   APL  ANES   PMB    INT  KINK  DISC
## 1   HIGH ZERO HIGH NORMAL FALSE FALSE FALSE FALSE FALSE NORMAL FALSE  TRUE
## 2   HIGH ZERO ZERO NORMAL FALSE FALSE FALSE FALSE FALSE NORMAL FALSE FALSE
## 3 NORMAL ZERO ZERO NORMAL FALSE FALSE FALSE FALSE FALSE NORMAL FALSE FALSE
## 4   HIGH ZERO ZERO NORMAL FALSE FALSE FALSE FALSE FALSE NORMAL FALSE FALSE
## 5    LOW ZERO ZERO NORMAL FALSE FALSE FALSE FALSE FALSE NORMAL FALSE FALSE
## 6   HIGH HIGH ZERO NORMAL FALSE FALSE FALSE  TRUE FALSE NORMAL FALSE FALSE
##      LVV   STKV CCHL  ERLO   HR  ERCA   SHNT    PVS   ACO2 VALV VLNG VTUB
## 1 NORMAL NORMAL HIGH FALSE HIGH FALSE NORMAL NORMAL NORMAL HIGH  LOW ZERO
## 2 NORMAL    LOW HIGH FALSE HIGH FALSE NORMAL    LOW    LOW ZERO ZERO  LOW
## 3 NORMAL NORMAL HIGH FALSE HIGH FALSE NORMAL    LOW    LOW ZERO ZERO  LOW
## 4 NORMAL NORMAL HIGH FALSE HIGH FALSE NORMAL NORMAL    LOW ZERO ZERO  LOW
## 5 NORMAL NORMAL HIGH FALSE HIGH FALSE NORMAL    LOW    LOW ZERO ZERO  LOW
## 6 NORMAL NORMAL HIGH FALSE HIGH FALSE NORMAL    LOW    LOW ZERO ZERO  LOW
##     VMCH
## 1 NORMAL
## 2 NORMAL
## 3 NORMAL
## 4 NORMAL
## 5 NORMAL
## 6 NORMAL
```

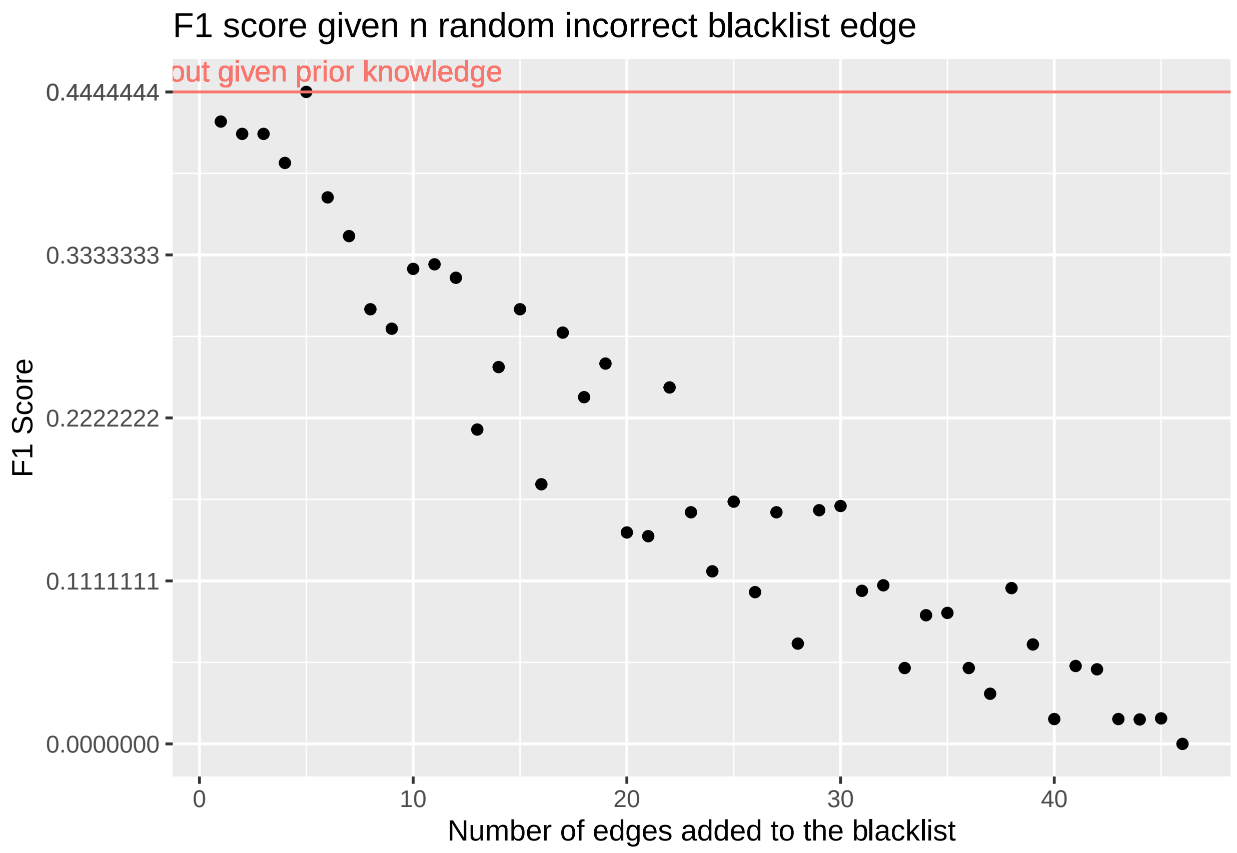
```
# Ground truth network
modelstring <- paste0("[HIST|LVF][CVP|LVV][PCWP|LVV][HYP][LVV|HYP:LVF][LVF]",
                      "[STKV|HYP:LVF][ERLO][HRBP|ERLO:HR][HREK|ERCA:HR][ERCA][HRSA|ERCA:HR][ANES]",
                      "[APL][TPR|APL][ECO2|ACO2:VLNG][KINK][MINV|INT:VLNG][FIO2][PVS|FIO2:VALV]",
                      "[SAO2|PVS:SHNT][PAP|PMB][PMB][SHNT|INT:PMB][INT][PRSS|INT:KINK:VTUB][DISC]",
                      "[MVS][VMCH|MVS][VTUB|DISC:VMCH][VLNG|INT:KINK:VTUB][VALV|INT:VLNG]",
                      "[ACO2|VALV][CCHL|ACO2:ANES:SAO2:TPR][HR|CCHL][CO|HR:STKV][BP|CO:TPR]")
dag_true <- model2network(modelstring)
graphviz.plot(dag_true)
```



```
# Given 1 incorrect edge to blacklist
n <- dim(dag_true$arcs)[1]
arcs <- dag_true$arcs
df_b1 <- data.frame(edge=character(), f1=numeric())

for (i in 1:n) {
  e <- arcs[i,]
  net <- hc(alarm, blacklist = e)

  df_b1[i,] <- c(paste(e, collapse = ' -> '), f1(compare(dag_true, net)))
}
df_b1$f1 = as.numeric(df_b1$f1)

gt_f1 = f1(compare(dag_true, hc(alarm)))

ggplot(df_b1, aes(x=edge, y=f1, group=1)) +
  scale_x_discrete(limits=df_b1$edge) +
  scale_y_continuous(breaks = sort(c(seq(min(df_b1$f1), max(df_b1$f1), length.out=5), gt_f1))) +
  geom_point() +
  geom_hline(aes(yintercept=gt_f1, color='red')) +
  geom_text(aes(5,gt_f1,label = 'Without given prior knowledge', vjust = -0.5, color='red')) +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
  labs(title = "F1 score given 1 incorrect blacklist edge",
       x='Edge added to the blacklist', y='F1 Score') +
  theme(legend.position = "none")
```

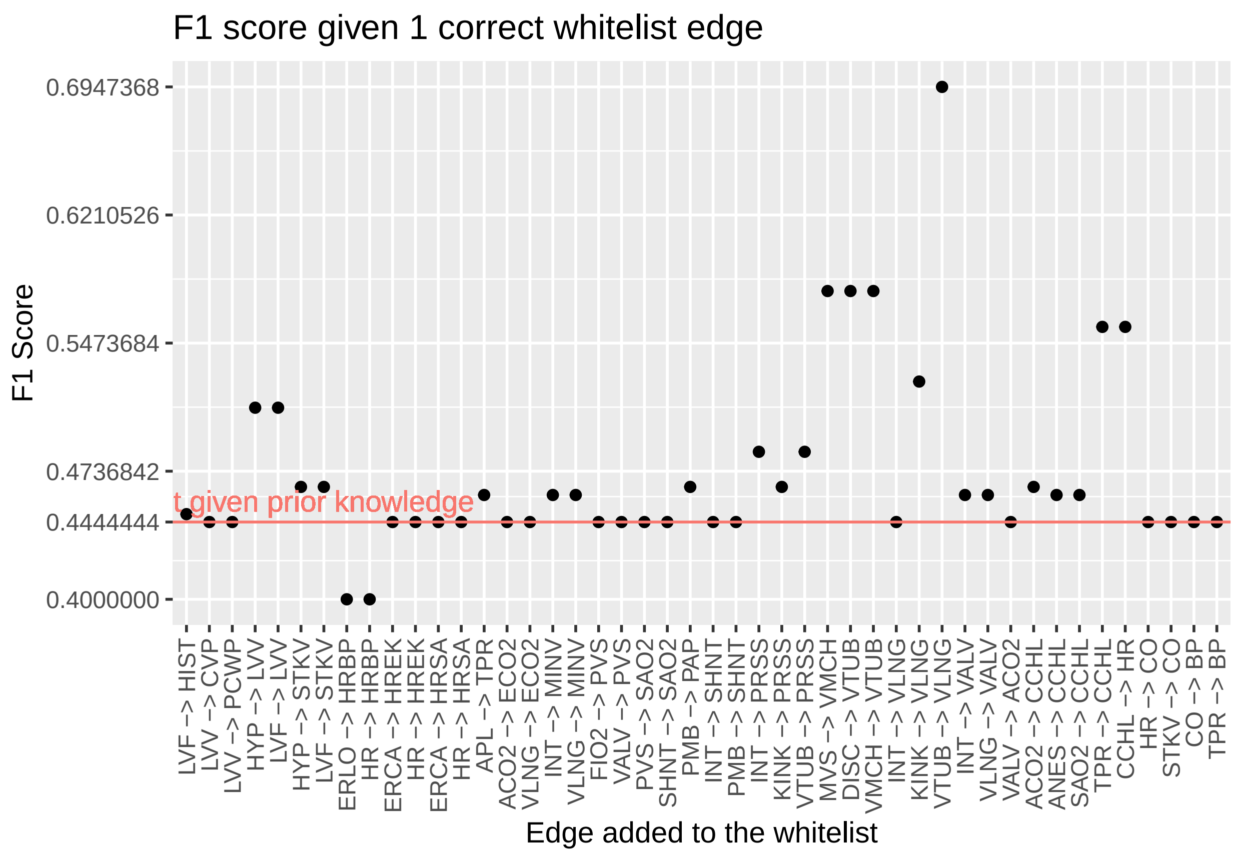


```
ggsave(
  'blacklist_1_f1.png',
  device = 'png',
  path = 'figures',
  width = 32,
  height = 18,
  units = 'cm'
)

# Given n random incorrect edge to blacklist
df_bn <- data.frame(f1=numeric())

for (i in 1:n) {
  e <- arcs[sample(1:n, i),]
  net <- hc(alarm, blacklist = e)

  df_bn[i,] <- f1(compare(dag_true, net))
}

ggplot(df_bn, aes(x=(1:n), y=f1)) +
  scale_y_continuous(breaks = sort(c(seq(min(df_bn$f1), max(df_bn$f1), length.out=5), gt_f1))) +
  geom_point() +
  geom_hline(aes(yintercept=gt_f1, color='red')) +
  geom_text(aes(5,gt_f1,label = 'Without given prior knowledge', vjust = -0.5, color='red')) +
  labs(title = "F1 score given n random incorrect blacklist edge",
       x='Number of edges added to the blacklist', y='F1 Score') +
  theme(legend.position = "none")
```



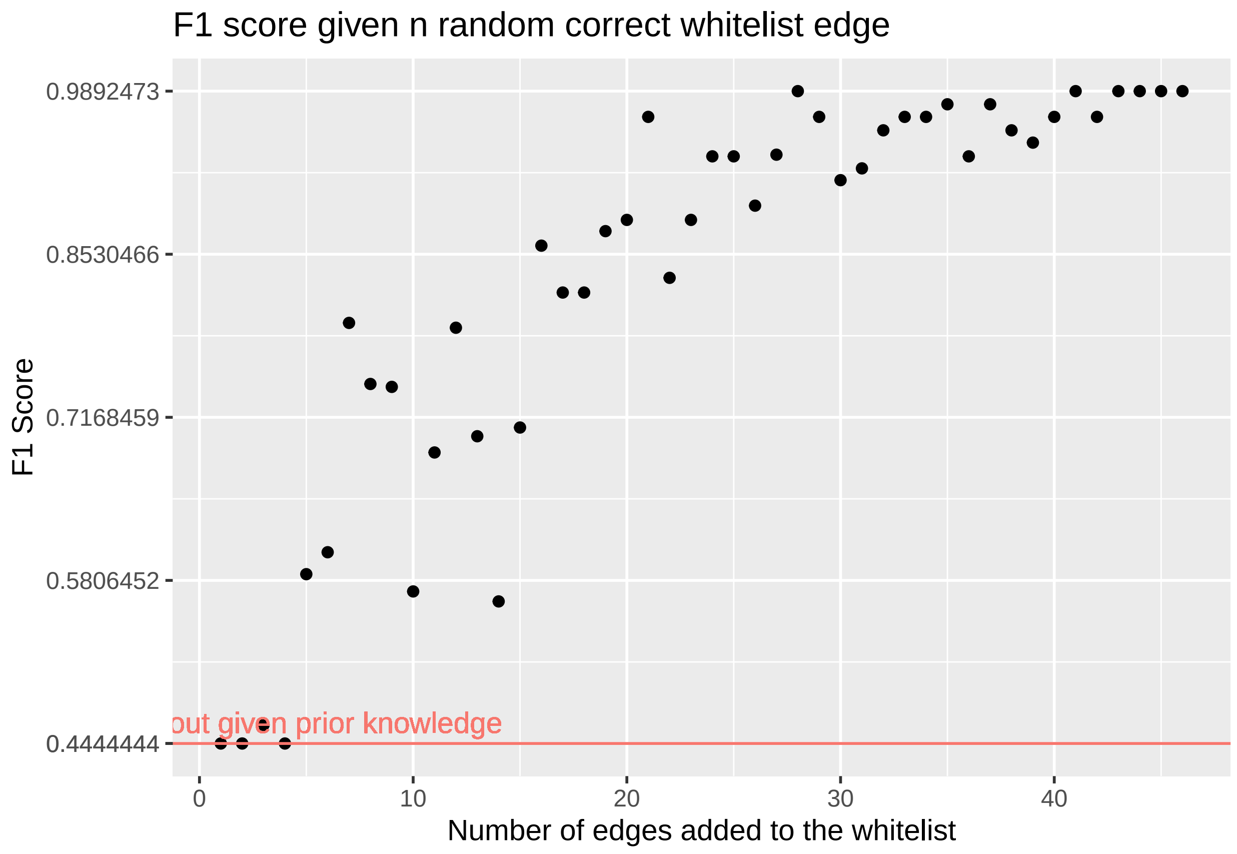
```
ggsave(
  'blacklist_n_f1.png',
  device = 'png',
  path = 'figures',
  width = 32,
  height = 18,
  units = 'cm'
)


# Given 1 correct edge to the white list
df_cw1 <- data.frame(edge=character(), f1=numeric())

for (i in 1:n) {
  e <- arcs[i,]
  net <- hc(alarm, whitelist = e)

  df_cw1[i,] <- c(paste(e, collapse = ' -> '), f1(compare(dag_true, net)))
}
df_cw1$f1 = as.numeric(df_cw1$f1)
```

```
ggplot(df_cw1, aes(x=edge, y=f1, group=1)) +
  scale_x_discrete(limits=df_cw1$edge) +
  scale_y_continuous(breaks = sort(c(seq(min(df_cw1$f1), max(df_cw1$f1), length.out=5), gt_f1))) +
  geom_point() +
  geom_hline(aes(yintercept=gt_f1, color='red')) +
  geom_text(aes(5,gt_f1,label = 'Without given prior knowledge', vjust = -0.5, color='red')) +
  theme(axis.text.x = element_text(angle = 90, vjust = 0.5, hjust=1)) +
  labs(title = "F1 score given 1 correct whitelist edge",
       x='Edge added to the whitelist', y='F1 Score') +
  theme(legend.position = "none")
```



```
ggsave(
  'whitelist_c1_f1.png',
  device = 'png',
  path = 'figures',
  width = 32,
  height = 18,
  units = 'cm'
)


# Given n random correct edge to the white list
df_cwn <- data.frame(f1=numeric())

for (i in 1:n) {
  e <- arcs[sample(1:n, i),]
```

```r
  net <- hc(alarm, whitelist = e)

  df_cwn[i,] <- f1(compare(dag_true, net))
}

ggplot(df_cwn, aes(x=(1:n), y=f1)) +
  scale_y_continuous(breaks = sort(c(seq(min(df_cwn$f1), max(df_cwn$f1), length.out=5), gt_f1))) +
  geom_point() +
  geom_hline(aes(yintercept=gt_f1, color='red')) +
  geom_text(aes(5,gt_f1,label = 'Without given prior knowledge', vjust = -0.5, color='red')) +
  labs(title = "F1 score given n random correct whitelist edge",
       x='Number of edges added to the whitelist', y='F1 Score') +
  theme(legend.position = "none")
```

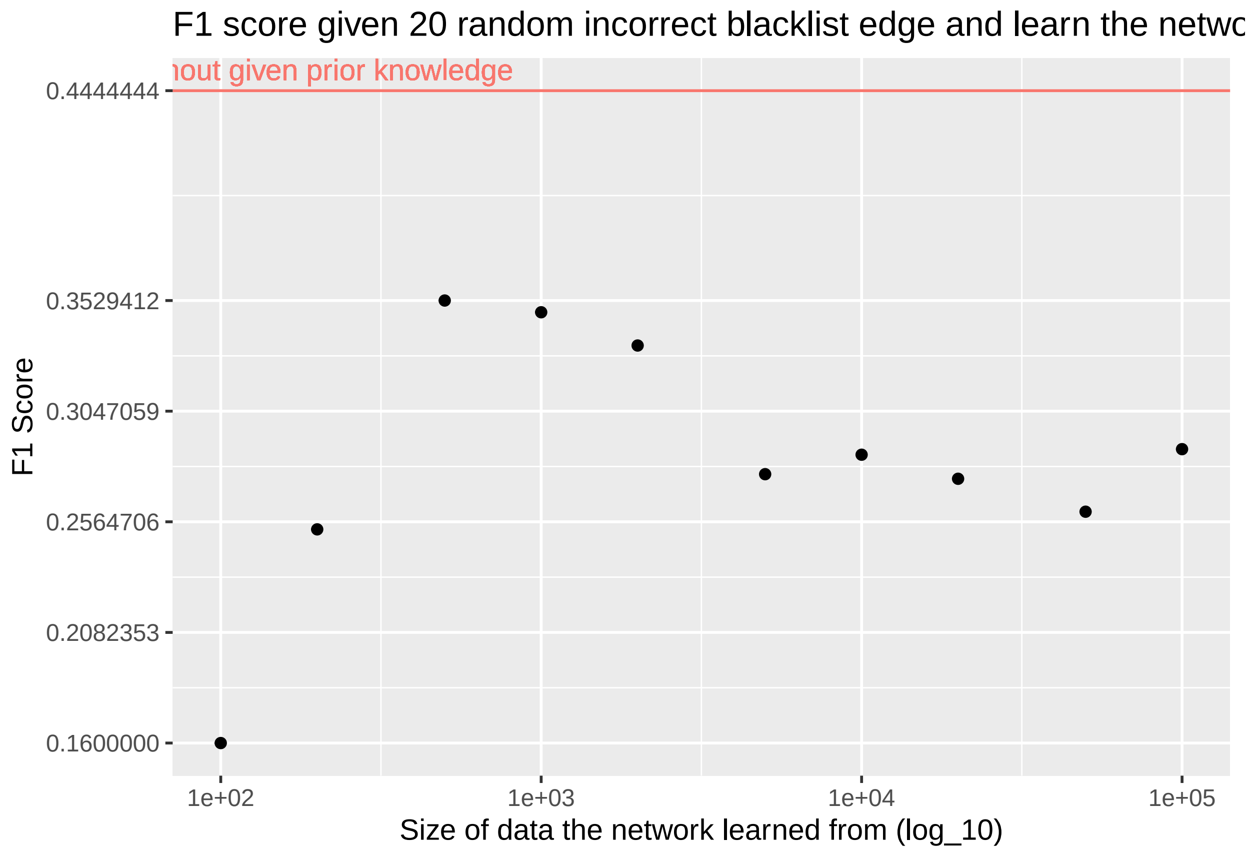


```r
ggsave(
  'whitelist_cn_f1.png',
  device = 'png',
  path = 'figures',
  width = 32,
  height = 18,
  units = 'cm'
)


# Given 20 random incorrect edge to the blacklist and learn the network from different size of data
df_size_b20 <- data.frame(size=numeric(), f1=numeric())
```

```
e <- arcs[sample(1:n, 20),]
for (i in c(100, 200, 500, 1000, 2000, 5000, 10000, 20000, 50000, 100000)) {
  sim <- rbn(dag_true, i, alarm, replace.unidentifiable = TRUE)

  net <- hc(sim, blacklist = e)

  df_size_b20[dim(df_size_b20)[1]+1,] <- c(i, f1(compare(dag_true, net)))
}
```

```
## Warning in check.data(x): variable VLNG has levels that are not observed in the
## data.
```

```
df_size_b20$size = as.numeric(df_size_b20$size)
df_size_b20$f1 = as.numeric(df_size_b20$f1)

ggplot(df_size_b20, aes(x=size, y=f1)) +
  scale_x_continuous(trans='log10') +
  scale_y_continuous(breaks = sort(c(seq(min(df_size_b20$f1), max(df_size_b20$f1), length.out=5), gt_f1
  geom_point() +
  geom_hline(aes(yintercept=gt_f1, color='red')) +
  geom_text(aes(200,gt_f1,label = 'Without given prior knowledge', vjust = -0.5, color='red')) +
  labs(title = "F1 score given 20 random incorrect blacklist edge and learn the network from different s
       x='Size of data the network learned from (log_10)', y='F1 Score') +
  theme(legend.position = "none")
```



```
ggsave(
  'blacklist_size_20_f1.png',
  device = 'png',
  path = 'figures',
  width = 32,
  height = 18,
  units = 'cm'
)



save.image('data.RData')
```