PrOCoil 2.0: R Example Code

Ulrich Bodenhofer and Annette Jacyszyn*

June 24, 2016

The results below are generated from an R script.

```
## to run this file as a whole and produce a PDF report, enter the following:
## > install.packages("knitr") ## if not already installed
## > library(knitr)
## > stitch("PrOCoil_Example_Code_V2.R")
## this produces a file PrOCoil_Example_Code_V2.tex which is then compiled
## to a PDF file; to compile the PDF, you need LaTeX installed on your
## computer.


## load packages
library(kebabs)
library(procoil)


## check version of 'procoil' package:
vers <- as.numeric(unlist(strsplit(packageDescription("procoil")$Version, "\\.")))
if (vers[1] < 2) stop("at least version 2.0.0 of the 'procoil' package is required")



## definition of function for augmenting PDB data
augmentDataSet <- function(PDB, BLAST, mapping)
{
    ## input checks
    if (!is(PDB, "AAStringSet") || !is(BLAST, "AAStringSet"))
        stop("'PDB' and 'BLAST' need to be 'AAStringSet' objects")

    ## perform mapping
    IDs <- unlist(mapping[names(PDB)])

    ## select mapped BLAST sequences
    out <- BLAST[IDs]

    ## make unique (multiple PDB sequences may be mapped to the same
    ## BLAST sequence)
    str <- paste0(as.character(out),
                  as.character(annotationMetadata(out)),
                  as.character(mcols(out)$Class))
    ustr <- unique(str)
```

*This report is automatically generated with the R package **knitr** (version 1.13).

```
    out <- out[match(ustr, str)]

    ## perform merge
    out <- c(PDB, out)

    ## add required metadata
    metadata(out)$annotationCharset <- "abcdefg"

    ## return final result
    out
}


## definition of function for creating a 'CCModel' object from a KeBABS model
createCCModel <- function(model)
{
    ## input checks
    if (!is(model, "KBModel"))
        stop("'model' needs to be 'KBModel' object")
    if (!is(model@svmInfo@reqKernel, "GappyPairKernel"))
        stop("'model' does not use the coiled coil kernel")

    ## extract feature weights
    weights <- getFeatureWeights(model)

    ## sort, arrange and properly name feature weights
    sel <- order(weights[1, ], decreasing=TRUE)
    weights <- weights[, sel, drop=FALSE]

    ## create and return final 'CCModel' object
    new("CCModel",
        b=model@b,
        m=as.integer(model@svmInfo@reqKernel@m),
        scaling=model@svmInfo@reqKernel@normalized,
        weights=weights)
}


## load PDB data set directly from the Web
prefix <- "http://www.bioinf.jku.at/software/procoil/data_v2/"
con <- url(paste0(prefix, "PrOCoil_PDB_V2.RData"))
load(con)
close(con)
## if you have 'PrOCoil_PDB_V2.RData' available locally,
## load it directly with
## > load("PrOCoil_PDB_V2.RData")

PDBdataX

##   A AAStringSet instance of length 1764
##        width seq                                                  names
##    [1]   271 LKLDKENALDRAEQAEADKKAAEDRSK...LEDELYAQKLKYKAISEELDHALNDM PDB1
##    [2]   271 LKLDKENALDRAEQAEADKKAAEDRSK...LEDELYAQKLKYKAISEELDHALNDM PDB2
##    [3]   253 KKAAEDRSKQLEDELVSLQKKLKGTED...LEDELYAQKLKYKAISEELDHALNDM PDB3
```

```
##     [4]   148 LEDKVEELLSKNYHLENEVARLKKLLE...EMKQLEDKVEELLSKNYHLENEVARL PDB4
##     [5]   148 LEDKVEELLSKNYHLENEVARLKKLLE...EMKQLEDKVEELLSKNYHLENEVARL PDB5
##     ...   ... ...
## [1760]    11 KEKLKELIFEE                                               PDB1760
## [1761]    11 LGLAHEALAAI                                               PDB1761
## [1762]    11 NEHLQKENERL                                               PDB1762
## [1763]    11 RNAVRALKSLS                                               PDB1763
## [1764]    11 TQKEAAWAITN                                               PDB1764

colnames(mcols(PDBdataX))

## [1] "PDB_IDs"    "Class"      "Fold"       "annotation"

## load BLAST data set directly from the Web
con <- url(paste0(prefix, "PrOCoil_BLAST_V2.RData"))
load(con)
close(con)
## if you have 'PrOCoil_BLAST_V2.RData' available locally,
## load it directly with
## > load("PrOCoil_BLAST_V2.RData")

BLASTdataX

##   A AAStringSet instance of length 1880
##         width seq                                                  names
##     [1]   110 FEELQDLCRQLHARVDKVDEERYDVEA...KQVKKEDIEKENREVGDWRKNIDALS BLAST1
##     [2]   110 FEELQDLCRQLHARVDKVDEERYDVEA...KQVKKEDIEKENREVGDWRKNIDALS BLAST2
##     [3]   110 SFAELQDLCRQLHARVDKVDEERYDVE...LKQVKKEDTEKENREVGDWRKNIDAL BLAST3
##     [4]   109 FAELQDLCRELHARVDKVDEERYDVEA...LKQVKKEDTEKENREVGDWRKNIDAL BLAST4
##     [5]   109 VAELQDLCRQLHARVDKVDEERYDVEA...LKQVKKEDTEKENREVGDWRKNIDAL BLAST5
##     ...   ... ...
## [1876]    19 VSRQRQEIGELRKEVEELS                                       BLAST1876
## [1877]    19 MRIEEMHKRLSKLEKKLDQ                                       BLAST1877
## [1878]    19 LKIDDLSKRIKALERKIKS                                       BLAST1878
## [1879]    18 GRQRQEILELRREMEELS                                        BLAST1879
## [1880]    17 RQRQEILELRREMEELS                                         BLAST1880

colnames(mcols(BLASTdataX))

## [1] "PDB_IDs"    "Class"      "Fold"       "annotation"

## load PDB -> BLAST mappings directly from the Web
con <- url(paste0(prefix, "PrOCoil_Augmentation_V2.RData"))
load(con)
close(con)
## if you have 'PrOCoil_Augmentation_V2.RData' available locally,
## load it directly with
## > load("PrOCoil_Augmentation_V2.RData")

str(PDB2BLASTmapping)

## List of 100
##  $ PDB58  : chr "BLAST84"
##  $ PDB69  : chr [1:4] "BLAST71" "BLAST72" "BLAST73" "BLAST74"
##  $ PDB78  : chr [1:5] "BLAST125" "BLAST137" "BLAST139" "BLAST146" ...
```

```
##  $ PDB99  : chr [1:2] "BLAST137" "BLAST1221"
##  $ PDB111 : chr "BLAST20"
##  $ PDB128 : chr [1:8] "BLAST79" "BLAST90" "BLAST196" "BLAST285" ...
##  $ PDB129 : chr [1:2] "BLAST117" "BLAST118"
##  $ PDB159 : chr [1:53] "BLAST1" "BLAST2" "BLAST3" "BLAST4" ...
##  $ PDB166 : chr [1:21] "BLAST44" "BLAST48" "BLAST49" "BLAST50" ...
##  $ PDB200 : chr [1:34] "BLAST365" "BLAST370" "BLAST394" "BLAST397" ...
##  $ PDB206 : chr [1:78] "BLAST34" "BLAST52" "BLAST53" "BLAST466" ...
##  $ PDB212 : chr [1:5] "BLAST23" "BLAST100" "BLAST104" "BLAST994" ...
##  $ PDB216 : chr [1:69] "BLAST474" "BLAST532" "BLAST564" "BLAST566" ...
##  $ PDB243 : chr [1:6] "BLAST47" "BLAST75" "BLAST80" "BLAST542" ...
##  $ PDB244 : chr "BLAST42"
##  $ PDB256 : chr "BLAST142"
##  $ PDB291 : chr [1:37] "BLAST78" "BLAST123" "BLAST548" "BLAST919" ...
##  $ PDB297 : chr [1:15] "BLAST119" "BLAST126" "BLAST177" "BLAST313" ...
##  $ PDB298 : chr [1:16] "BLAST501" "BLAST530" "BLAST531" "BLAST536" ...
##  $ PDB306 : chr [1:15] "BLAST86" "BLAST87" "BLAST91" "BLAST95" ...
##  $ PDB308 : chr "BLAST64"
##  $ PDB324 : chr [1:24] "BLAST1386" "BLAST1391" "BLAST1394" "BLAST1396" ...
##  $ PDB340 : chr [1:58] "BLAST122" "BLAST155" "BLAST171" "BLAST352" ...
##  $ PDB349 : chr "BLAST1493"
##  $ PDB371 : chr [1:102] "BLAST70" "BLAST76" "BLAST77" "BLAST120" ...
##  $ PDB378 : chr [1:122] "BLAST81" "BLAST88" "BLAST89" "BLAST92" ...
##  $ PDB396 : chr [1:25] "BLAST590" "BLAST657" "BLAST749" "BLAST754" ...
##  $ PDB405 : chr [1:2] "BLAST1575" "BLAST1737"
##  $ PDB406 : chr [1:28] "BLAST86" "BLAST87" "BLAST91" "BLAST95" ...
##  $ PDB428 : chr [1:89] "BLAST1288" "BLAST1290" "BLAST1295" "BLAST1298" ...
##  $ PDB459 : chr [1:7] "BLAST193" "BLAST278" "BLAST405" "BLAST441" ...
##  $ PDB474 : chr [1:50] "BLAST19" "BLAST21" "BLAST24" "BLAST25" ...
##  $ PDB479 : chr [1:2] "BLAST699" "BLAST787"
##  $ PDB482 : chr [1:125] "BLAST81" "BLAST85" "BLAST88" "BLAST89" ...
##  $ PDB483 : chr [1:45] "BLAST462" "BLAST473" "BLAST616" "BLAST643" ...
##  $ PDB489 : chr [1:5] "BLAST608" "BLAST617" "BLAST622" "BLAST624" ...
##  $ PDB502 : chr [1:150] "BLAST153" "BLAST154" "BLAST156" "BLAST158" ...
##  $ PDB528 : chr [1:46] "BLAST454" "BLAST459" "BLAST492" "BLAST529" ...
##  $ PDB535 : chr [1:106] "BLAST209" "BLAST211" "BLAST376" "BLAST461" ...
##  $ PDB552 : chr [1:3] "BLAST596" "BLAST851" "BLAST870"
##  $ PDB557 : chr "BLAST1814"
##  $ PDB562 : chr [1:48] "BLAST478" "BLAST483" "BLAST499" "BLAST501" ...
##  $ PDB566 : chr [1:8] "BLAST1510" "BLAST1687" "BLAST1711" "BLAST1724" ...
##  $ PDB575 : chr [1:91] "BLAST884" "BLAST1437" "BLAST1439" "BLAST1442" ...
##  $ PDB591 : chr [1:21] "BLAST855" "BLAST865" "BLAST936" "BLAST1292" ...
##  $ PDB596 : chr [1:52] "BLAST1664" "BLAST1665" "BLAST1681" "BLAST1682" ...
##  $ PDB597 : chr [1:102] "BLAST70" "BLAST76" "BLAST836" "BLAST912" ...
##  $ PDB607 : chr [1:50] "BLAST19" "BLAST21" "BLAST24" "BLAST25" ...
##  $ PDB620 : chr [1:2] "BLAST1721" "BLAST1807"
##  $ PDB622 : chr [1:124] "BLAST70" "BLAST76" "BLAST77" "BLAST120" ...
##  $ PDB628 : chr "BLAST64"
##  $ PDB630 : chr [1:79] "BLAST592" "BLAST593" "BLAST625" "BLAST652" ...
##  $ PDB639 : chr "BLAST372"
##  $ PDB642 : chr "BLAST64"
##  $ PDB647 : chr "BLAST64"
##  $ PDB656 : chr [1:16] "BLAST86" "BLAST87" "BLAST91" "BLAST95" ...
```

```
##  $ PDB659 : chr [1:73] "BLAST204" "BLAST207" "BLAST214" "BLAST219" ...
##  $ PDB664 : chr "BLAST64"
##  $ PDB667 : chr [1:51] "BLAST589" "BLAST599" "BLAST601" "BLAST605" ...
##  $ PDB670 : chr "BLAST20"
##  $ PDB693 : chr "BLAST1496"
##  $ PDB709 : chr "BLAST1731"
##  $ PDB711 : chr [1:10] "BLAST868" "BLAST1102" "BLAST1369" "BLAST1397" ...
##  $ PDB718 : chr [1:19] "BLAST904" "BLAST906" "BLAST921" "BLAST929" ...
##  $ PDB726 : chr [1:2] "BLAST1765" "BLAST1782"
##  $ PDB734 : chr [1:8] "BLAST1405" "BLAST1494" "BLAST1511" "BLAST1620" ...
##  $ PDB735 : chr [1:45] "BLAST1681" "BLAST1691" "BLAST1705" "BLAST1707" ...
##  $ PDB754 : chr "BLAST20"
##  $ PDB755 : chr [1:28] "BLAST86" "BLAST87" "BLAST91" "BLAST95" ...
##  $ PDB778 : chr [1:28] "BLAST1350" "BLAST1416" "BLAST1418" "BLAST1487" ...
##  $ PDB779 : chr [1:2] "BLAST1765" "BLAST1782"
##  $ PDB781 : chr [1:50] "BLAST1433" "BLAST1435" "BLAST1444" "BLAST1457" ...
##  $ PDB815 : chr [1:5] "BLAST1326" "BLAST1365" "BLAST1368" "BLAST1419" ...
##  $ PDB830 : chr [1:2] "BLAST1291" "BLAST1563"
##  $ PDB859 : chr [1:6] "BLAST22" "BLAST46" "BLAST579" "BLAST628" ...
##  $ PDB892 : chr [1:10] "BLAST1486" "BLAST1490" "BLAST1509" "BLAST1528" ...
##  $ PDB911 : chr "BLAST1811"
##  $ PDB912 : chr [1:2] "BLAST1645" "BLAST1648"
##  $ PDB921 : chr [1:37] "BLAST1681" "BLAST1705" "BLAST1716" "BLAST1725" ...
##  $ PDB927 : chr [1:10] "BLAST1405" "BLAST1431" "BLAST1494" "BLAST1511" ...
##  $ PDB953 : chr [1:16] "BLAST1187" "BLAST1703" "BLAST1738" "BLAST1745" ...
##  $ PDB961 : chr "BLAST1829"
##  $ PDB987 : chr [1:43] "BLAST1289" "BLAST1311" "BLAST1315" "BLAST1340" ...
##  $ PDB1001: chr "BLAST1492"
##  $ PDB1055: chr [1:22] "BLAST1386" "BLAST1391" "BLAST1394" "BLAST1396" ...
##  $ PDB1068: chr [1:52] "BLAST873" "BLAST882" "BLAST891" "BLAST898" ...
##  $ PDB1074: chr [1:4] "BLAST864" "BLAST899" "BLAST914" "BLAST1821"
##  $ PDB1076: chr [1:46] "BLAST552" "BLAST572" "BLAST582" "BLAST586" ...
##  $ PDB1104: chr "BLAST64"
##  $ PDB1136: chr [1:17] "BLAST1187" "BLAST1703" "BLAST1738" "BLAST1745" ...
##  $ PDB1152: chr "BLAST1811"
##  $ PDB1202: chr [1:19] "BLAST1708" "BLAST1855" "BLAST1861" "BLAST1862" ...
##  $ PDB1442: chr [1:7] "BLAST22" "BLAST23" "BLAST46" "BLAST100" ...
##  $ PDB1631: chr [1:37] "BLAST1764" "BLAST1783" "BLAST1787" "BLAST1790" ...
##  $ PDB1633: chr [1:43] "BLAST1289" "BLAST1311" "BLAST1315" "BLAST1340" ...
##  $ PDB1637: chr [1:5] "BLAST142" "BLAST1736" "BLAST1743" "BLAST1750" ...
##  $ PDB1724: chr [1:2] "BLAST615" "BLAST826"
##  $ PDB1737: chr [1:2] "BLAST1291" "BLAST1563"
##  $ PDB1738: chr [1:11] "BLAST1405" "BLAST1431" "BLAST1494" "BLAST1511" ...
##   [list output truncated]

## define coiled coil kernel with m = 5
CCkernel5 <- gappyPairKernel(k=1, m=5, normalize=TRUE, annSpec=TRUE)


## example showing how to perform grouped cross validation on the PDB data set
## with the folds defined in the metadata column 'Fold'
res <- kbsvm(x=PDBdataX, y=mcols(PDBdataX)$Class, kernel=CCkernel5,
             svm="C-svc", pkg="Liblinear", explicit="yes", cross=10,
             groupBy=mcols(PDBdataX)$Fold, cost=2)
```

```
cvResult(res)
```

```
##
## Cross validation result object of class  "CrossValidationResult"
##
## cross                 : 10
## noCross               : 1
##
## CV error:             : 0.12842529
```

```
## example that trains a model on all folds except no. 8 (training set
## augmented with BLAST sequences) and finally makes a prediction on
## fold no. 8 as test set (PDB sequences only)

## select samples
testFold <- which(mcols(PDBdataX)$Fold == 8)
trainSet <- PDBdataX[-testFold]
testSet <- PDBdataX[testFold]

## augment training set
trainSet <- augmentDataSet(trainSet, BLASTdataX, PDB2BLASTmapping)
trainSet
```

```
##   A AAStringSet instance of length 3209
##        width seq                                                  names
##    [1]   271 LKLDKENALDRAEQAEADKKAAEDRSK...LEDELYAQKLKYKAISEELDHALNDM PDB1
##    [2]   271 LKLDKENALDRAEQAEADKKAAEDRSK...LEDELYAQKLKYKAISEELDHALNDM PDB2
##    [3]   253 KKAAEDRSKQLEDELVSLQKKLKGTED...LEDELYAQKLKYKAISEELDHALNDM PDB3
##    [4]   148 LEDKVEELLSKNYHLENEVARLKKLLE...EMKQLEDKVEELLSKNYHLENEVARL PDB4
##    [5]   148 LEDKVEELLSKNYHLENEVARLKKLLE...EMKQLEDKVEELLSKNYHLENEVARL PDB5
##     ...   ... ...
## [3205]    28 SSQELAELKKQVESAELKNQRLREVFQT                            BLAST1750
## [3206]    23 NIAELHQLREECERLRELVRVLE                                 BLAST1849
## [3207]    53 VLKYKIRKKKAEHKLVETDENLYRVLDILHELDNRLEPLEMQASSARDYVQMS   BLAST615
## [3208]    50 KYKIRKKKAEHKLVETDENLYRVLDILHELDSRLGPLEMQASSARDYVQM      BLAST826
## [3209]    23 SKDNELKNLKERCKILEEKLARY                                 BLAST1847
```

```
## train model
model <- kbsvm(x=trainSet, y=mcols(trainSet)$Class, kernel=CCkernel5,
               svm="C-svc", pkg="Liblinear", explicit="yes", cost=2)

## make prediction
pred <- predict(model, testSet)
evaluatePrediction(pred, mcols(testSet)$Class)
```

```
##        TRIMER DIMER
## TRIMER     13     3
## DIMER      10   143
##
## Accuracy:              92.308% (156 of 169)
## Balanced accuracy:     77.233% (13 of 23 and 143 of 146)
## Matthews CC:            0.638
##
## Sensitivity:           56.522% (13 of 23)
## Specificity:           97.945% (143 of 146)
## Precision:             81.250% (13 of 16)
```

```
## example how to train the final PrOCoil model

## augment entire PDB data set
mergedSet <- augmentDataSet(PDBdataX, BLASTdataX, PDB2BLASTmapping)
mergedSet

##   A AAStringSet instance of length 3644
##        width seq                                                  names
##    [1]   271 LKLDKENALDRAEQAEADKKAAEDRSK...LEDELYAQKLKYKAISEELDHALNDM PDB1
##    [2]   271 LKLDKENALDRAEQAEADKKAAEDRSK...LEDELYAQKLKYKAISEELDHALNDM PDB2
##    [3]   253 KKAAEDRSKQLEDELVSLQKKLKGTED...LEDELYAQKLKYKAISEELDHALNDM PDB3
##    [4]   148 LEDKVEELLSKNYHLENEVARLKKLLE...EMKQLEDKVEELLSKNYHLENEVARL PDB4
##    [5]   148 LEDKVEELLSKNYHLENEVARLKKLLE...EMKQLEDKVEELLSKNYHLENEVARL PDB5
##    ...   ... ...
## [3640]    28 SSQELAELKKQVESAELKNQRLREVFQT                            BLAST1750
## [3641]    23 NIAELHQLREECERLRELVRVLE                                 BLAST1849
## [3642]    53 VLKYKIRKKKAEHKLVETDENLYRVLDILHELDNRLEPLEMQASSARDYVQMS    BLAST615
## [3643]    50 KYKIRKKKAEHKLVETDENLYRVLDILHELDSRLGPLEMQASSARDYVQM       BLAST826
## [3644]    23 SKDNELKNLKERCKILEEKLARY                                 BLAST1847

## train model
model <- kbsvm(x=mergedSet, y=mcols(mergedSet)$Class, kernel=CCkernel5,
               svm="C-svc", pkg="Liblinear", explicit="yes", cost=2)

## convert to 'CCModel' object
pModel <- createCCModel(model)
pModel

## An object of class  "CCModel"
##
## Model parameters:
##  coiled coil kernel with m=5 and kernel normalization
##  offset b= -1.073
##
## Feature weights:
##    1.6351 ... L...Vd...a
##    1.5379 ... R....Eg....e
##    1.2894 ... R.Ec.e
##    1.2273 ... E..Ve..a
##    1.2039 ... I...Id...a
##       ... ... ...
##   -1.1327 ... K..La..d
##   -1.2203 ... E.Ec.e
##   -1.2304 ... L..Ld..g
##   -1.4274 ... L...Nd...a
##   -1.7826 ... N..La..d

## note that the SVM algorithm used above has a stochastic component,
## so the resulting model will not be identical to the published model
## 'PrOCoilModel', but it will be very similar and produce the same predictions

## compare with 'PrOCoilModel' contained in 'procoil' R package
PrOCoilModel

## An object of class  "CCModel"
```

```
##
## Model parameters:
##  coiled coil kernel with m=5 and kernel normalization
##  offset b= -1.073
##
## Feature weights:
##    1.6363 ... L...Vd...a
##    1.5382 ... R....Eg....e
##    1.2903 ... R.Ec.e
##    1.2284 ... E..Ve..a
##    1.2040 ... I...Id...a
##       ... ... ...
##   -1.1330 ... K..La..d
##   -1.2192 ... E.Ec.e
##   -1.2290 ... L..Ld..g
##   -1.4273 ... L...Nd...a
##   -1.7811 ... N..La..d
```

The R session information (including the OS info, R version and all packages used):

```
sessionInfo()

## R version 3.3.0 (2016-05-03)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Ubuntu 16.04 LTS
##
## locale:
##  [1] LC_CTYPE=en_US.UTF-8       LC_NUMERIC=C               LC_TIME=en_US.UTF-8
##  [4] LC_COLLATE=en_US.UTF-8     LC_MONETARY=en_US.UTF-8    LC_MESSAGES=en_US.UTF-8
##  [7] LC_PAPER=en_US.UTF-8       LC_NAME=C                  LC_ADDRESS=C
## [10] LC_TELEPHONE=C             LC_MEASUREMENT=en_US.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] stats4    parallel  stats     graphics  grDevices utils     datasets  methods
## [9] base
##
## other attached packages:
##  [1] LiblineaR_1.94-2    procoil_2.0.2       kebabs_1.6.2        kernlab_0.9-24
##  [5] Biostrings_2.40.2   XVector_0.12.0      IRanges_2.6.1       S4Vectors_0.10.1
##  [9] BiocGenerics_0.18.0 knitr_1.13
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_0.12.5     lattice_0.20-33 class_7.3-14    grid_3.3.0      formatR_1.4
##  [6] magrittr_1.5    e1071_1.6-7     evaluate_0.9    highr_0.6       stringi_1.1.1
## [11] zlibbioc_1.18.0 SparseM_1.7     Matrix_1.2-6    apcluster_1.4.3 tools_3.3.0
## [16] stringr_1.0.0

Sys.time()

## [1] "2016-06-24 13:39:43 CEST"
```