Published in final edited form as:

Nat Protoc. 2017 May ; 12(5): 1089–1102. doi:10.1038/nprot.2017.022.

Assessing engineered cells using CellNet and RNA-Seq

Arthur H Radley¹, Remy M Schwab^{1,2}, Yuqi Tan^{1,2}, Jeesoo Kim¹, Emily KW Lo^{1,2}, and Patrick Cahan^{1,2}

¹Department of Biomedical Engineering, Johns Hopkins University School of Medicine, Baltimore, Maryland, 21205 USA

²Institute for Cell Engineering, Johns Hopkins University School of Medicine, Baltimore, Maryland, 21205 USA

Abstract

CellNet is a computational platform designed to assess cell populations engineered via either directed differentiation of pluripotent stem cells or via direct conversion, and to suggest specific hypotheses to improve cell fate engineering protocols. CellNet takes as input gene expression data and compares it to large data sets of normal expression profiles compiled from public sources in terms of the extent to which cell and tissue specific gene regulatory networks are established. CellNet was originally designed to work with human or mouse microarray expression data of 21 cell and tissue types. Here we describe how to apply CellNet to RNA-Seq data and how to build a completely new CellNet platform applicable to, for example, other species or additional cell and tissue types. Once the raw data has been pre-processed, running CellNet only takes several minutes whereas the time required to create a completely new CellNet requires several hours.

Keywords

Gene regulatory networks; computational biology; stem cells; cell fate engineering; classification

Introduction

Development of the protocol

Cell fate engineering, for example the directed differentiation of pluripotent stem cells (PSC)¹ or the direct conversion among somatic cell types (e.g. conversion of fibroblasts to cardiomyocytes through the ectopic expression of Gata4, Mef2c, and Tbx5, ref. 2) is practiced in thousands of labs worldwide to model diseases, to explore inaccessible time points in development, to screen drugs, and to develop regenerative medicine therapies. There are several challenges to realizing the full potential of cell fate engineering for these

Correspondence to: Patrick Cahan, patrick.cahan@jhmi.edu, 410-614-7827, <http://www.cahanlab.org/>.

TWEET: Applying CellNet to RNA-Seq data to improve cell fate engineering protocols @cahanLab

Author contributions statements: AR wrote code, performed analysis, and wrote the manuscript. RS wrote code, and performed analysis. YT analyzed data, debugged code, and edited the manuscript. JK debugged code and analyzed data. EKW analyzed data and edited the manuscript. PC devised the method, wrote code, analyzed data, wrote the manuscript, and oversaw the project.

Competing Financial interests: The authors have no competing financial interests.

purposes. First, the resemblance of engineered cells and populations to their *in vivo* counterparts is difficult to determine. While functional complementation via transplantation in live animals³ has been used to assess the ability of engineered cells to mimic the physiology of their native counterparts, such experiments are technically challenging, lack quantitative rigor, and provide limited insights when judging human tissue function in animal hosts. The molecular fidelity of engineered populations is typically assessed by semi-quantitative PCR⁴, array-based expression profiling⁵, or RNA sequencing⁶ followed by clustering analysis.

Second, deriving cell fate engineering protocols, either directed differentiation or direct conversion, has been less of an engineering task and more of an empirical trial and error task based on what we can glean from development or from comparative expression studies. Protocols to direct the differentiation of PSC to selected lineages are inspired by our understanding of signaling cues and mechanical forces that pattern the embryo and guide cell fate decisions¹. However, identifying these signals is limited by our inability to access transient stages during early development. On the other hand, direct conversion protocols are typically based on the identification of a set of lineage-specific master regulators, which are thought to auto-regulate expression, positively regulate the transcription of cell type-associated genes, and repress alternative lineages⁷. While this strategy appears to apply to reprogramming to pluripotency, the extent to which it applies to other cell types is unknown.

We previously developed a computational platform, CellNet, to address these two issues⁸. CellNet uses as its basis for comparison the gene regulatory networks (GRNs) of cell and tissue (C/T) types in human and mouse that we reconstructed from thousands of publicly available gene expression profiles. It takes as input gene expression data from cell fate engineering experiments, and produces three outputs (Figure 1): 1) a classification score indicating the extent to which a query sample is indistinguishable in its expression profile from each of the reference C/T types; 2) a metric of the extent to which a cell- or tissue-specific GRN is established in a query sample (GRN status); and 3) a list of transcription factors scored according to how likely their expression modulation would improve the desired fate change, which we refer to as the Network Influence Score (NIS).

By applying CellNet to gene expression data of compatible cell engineering experiments in the public domain, we answered several lingering and pressing questions in the field. First, we found that cells derived by directed differentiation resembled their *in vivo* target cell types more closely than those derived through direct conversion. Second, we found that the GRNs of the starting cell type frequently are maintained in cells engineered either through direct conversion or directed differentiation. Third, we documented the substantial improvement of target cell type GRN status when cell fate engineering was practiced *in situ*, or after engineered cells were transplanted into their native niche. Finally, we discovered the aberrant establishment of GRNs of other cell types (neither the starting nor the target) in engineered cells, an insight that led to the discovery of a colon/liver bi-potent endoderm progenitor resulting from direct conversion of fibroblasts towards hepatocyte fate⁹.

CellNet has been applied in diverse cell engineering contexts, including improved engineering of hepatocytes¹⁰⁻¹², direct conversion of fibroblasts to cardiomyocytes¹³, the

characterization of the maturation of engineered cardiomyocytes¹⁴, the functional improvement of directly converted macrophages⁹, and the detection of a multi-lineage primed state in engineered hematopoietic stem cells¹⁵.

The original version of CellNet was applied to microarray data⁸. Based on the recent widespread accessibility of RNA-Seq as a method for estimating gene expression, we additionally demonstrate here the use of CellNet to analyze RNA-Seq data. To increase accessibility of CellNet and its reproducible use, we have created an ‘image’ on Amazon's EC2 cloud on which we have installed all the software and R packages needed to follow this protocol. We have also provided example training and query data sets, as well as intermediate results for each step in the PROCEDURE so that the user can test each step. We note that the user will be charged for any costs associated with using Amazon Web Services.

Applications of the method

CellNet was designed primarily as a tool to aid in cell fate engineering, and most applications to date have been in this context. However, CellNet can be applied in any biological context where it would be informative to assess the status of tissue and cell type-specific GRNs and cell type identity. For example, one could use CellNet to predict the tissue of origin for metastatic cancers where the primary tumor is unknown¹⁶. More generally, CellNet could be applied to expression data of tumors to explore how normal GRNs are re-wired by tumorigenesis, and even more broadly, to explore how normal C/T regulatory networks are affected by other diseases or chronic states. In the future we will extend CellNet to single cell RNA-Seq. We have not yet assessed the ability of CellNet to estimate the relative contribution of C/Ts to populations of mixed composition.

Comparison with other methods

Traditionally, the molecular profiles of engineered populations have been compared to the starting and target C/T types using unsupervised approaches such as principal component analysis or hierarchical clustering (HCL). We have showed that classification by HCL has lower precision as compared to CellNet classification⁸. More importantly, by incorporating the GRNs of many C/T types, CellNet is able to detect the aberrant activation of alternative C/T GRNs (i.e. those that are associated with neither starting or target C/T type). Several new approaches to assess and/or suggest improvements to cell fate engineering protocols have been described, which we recently reviewed¹⁷. ScoreCard assesses the lineage propensity of nominally pluripotent populations⁴, and PluriTest assesses the resemblance of such populations to embryonic stem cells¹⁸. TeratoScore compares profiles of teratomas to those of mature cell types as a quantitative readout on the teratoma assay of pluripotency¹⁹, and KeyGenes compares profiles of engineered populations to those of fetal tissues²⁰. There are several computational approaches that can be used to suggest improvements to cell fate engineering. Only a few of these approaches have been prospectively tested, including Mogrify²¹, which also uses GRNs to prioritize sets of transcription factors, and the approach of D'Alessio et al, which scores fate-defining factors based on specificity of expression²².

Limitations of the protocol

There are several limitations of the current protocol. The first limitation is that we require strictly adhering to our pre-processing pipeline, which consists of trimming reads and applying the RNA-Seq quantification tool, Salmon, to estimate expression, so that query expression profiles can be meaningfully compared to the training data. Second, the protocol is limited to those cell and tissue types that are publicly available as raw data. We have trained CellNet platforms based on RNA-Seq data from 16 human and 16 mouse C/T types, and we will continue to add more types as sufficient data becomes available. Finally, while a common goal in cell fate engineering is the derivation of a relatively homogenous population of a single cell type, this version of CellNet is trained on data from bulk populations or tissues rather than from single cells.

Level of Expertise Needed to Implement the Protocol

The step-by-step protocol and supporting information within this document are designed for intermediate to advanced users of the R programming language. Pre-processing raw RNA-Seq data requires the use of command-line tools such as Salmon and cutadapt. Therefore, experience using the shell and installing programs from source code is highly recommended.

Experimental Design

This PROCEDURE describes how to use CellNet to analyze RNA-Seq query data and how to construct CellNet (Figure 2). The steps listed here assume that the user is performing the analysis on Amazon's cloud service EC2 using the AMI image, the training data, and the query data that we have provided. Users will have to adjust some of the code to adapt the protocol to their own data (e.g. file names and R function call arguments).

If the user wishes to analyze query data and is not adding to or creating a new CellNet platform, then they only need to follow Steps 1-12. We illustrate these parts of the Procedure by analyzing a published time course of reprogramming to pluripotency using a doxycycline-inducible system in murine cells²³ and the directed differentiation of human induced pluripotent stem cells (iPSC) to hippocampal dentate gyrus granule neurons²⁴. We have provided FASTQ files, a metadata table, as well as pre-processed query data so that the user can walk through all of these steps. In order for the user to adapt the protocol for their data, they will have to create their own meta data table, upload their raw data, and change the files names listed in these sections.

If the user wants to add new cell or tissue types or to create a CellNet platform for a different species, then Steps 13 - 21 should be followed. The magnitude of the raw gene expression data used to train the human and mouse RNA-Seq versions of CellNet precludes us from making these FASTQ files for the training steps (Steps 13-16) available for download; however, we have provided the outputs from this step (normalized expression data) so that the user can follow the subsequent steps to reconstruct GRNs (Step 17) and train and assess CellNet (Steps 18-21).

Create query metadata file, Step 1

The user must create a comma separated values (.csv) file or an R data frame that contains metadata for each query sample. The metadata must include a unique sample identifier, the file name of the raw data, and an annotation (e.g. experimental group), which can be used to group the query samples in the CellNet output. Supplementary Table 1 is the csv version of the query metadata that we use in Step 2. The steps here are identical to those in the Step 13 under the Pre-process training data section except that here the sample information corresponds to samples that the user wishes to analyze with CellNet.

Pre-process query data, Steps 2 – 6

The RNA-Seq pre-processing steps convert raw reads into normalized read counts per gene. To standardize the analysis and to maximize our use of publicly available RNA-Seq data, we use only one end, if the data is from paired-end-read runs, and we trim reads to a length of 40 bases. We use cutadapt²⁵ to trim reads, and we use the quasi-mapping algorithm Salmon²⁶ to estimate abundances of transcripts that are defined in a user-provided FASTA file that lists the sequences of transcripts (see **EQUIPMENT SETUP**). Salmon converts the transcript-defining FASTA file into an index, which is used to perform the quasi-mapping. We have provided Salmon indices for the mouse and human transcriptomes, plus commonly used exogenous spike ins, reporters (e.g. eGFP), and selection (e.g. ampicillin) genes. To derive a read count per gene, we sum the counts per transcript across all transcripts associated with a gene. Both the indexing and the transcript-to-gene summarization are based on Ensembl-provided sequences and gene annotations (<ftp://ftp.ensembl.org/pub/release-80/gtf/>).

The same process is used to pre-process the training data as is used to pre-process the query data that is to be analyzed using CellNet. To make the raw expression data comparable across read depths, we down sample the raw quasi-mapped reads such that the total reads per sample is 100,000. Effectively, we perform the down-sampling by subtracting from the raw reads an average of the total number of reads – 100,000, weighted by the per-gene read count. Then we transform the expression by taking the natural logarithm of (1+downsampled read count). The result of this section is a transformed expression matrix. We also tested the size factors normalization approach of DESeq²⁷ but found the total counts normalization yielded better GRN performance as determined by comparison to ‘gold standard’ sets of transcription factor targets (Supplementary Figure 1).

Analyze query data with CellNet, Steps 7 – 12

Applying CellNet to the processed query data produces three outputs (Figure 1): (1) the classification score indicating the extent to which query samples are indistinguishable in their expression profile from each of the reference C/T types; (2) the GRN status, a metric of the extent to which a C/T specific GRN is established in a query sample; and (3) a list of transcription factors scored according to how likely their expression modulation would improve the desired fate change, which is called the Network Influence Score (NIS).

Pre-process training data, Steps 13 – 16

The steps here are identical to those in the ‘Pre-process query data’ section except that the raw data here is from samples that will be used to train a new CellNet. The result of this section is a normalized expression matrix. We provide pre-processed data that can be used as input the next section: ‘GRN reconstruction’.

One of the most arduous tasks is to define and ‘harvest’ sufficient raw expression data. Our approach to this task is to search public repositories of gene expression data for studies that include profiles of C/Ts of interest that are from healthy, wild-type, or perturbed but non-cancerous sources. For each study we manually create a metadata table that lists sufficient information to fetch the associated raw expression data files, which we save as a .csv file. Ultimately, we combine these metadata tables into a single comprehensive metadata R file. Supplementary Table 2 illustrates the column headers and the first several rows of the example metadata table that we provide. We do not provide FASTQ files of the training data. In Step 17 of the PROCEDURE we assume that the user has stored the FASTQ files of the training data on the computer where they will run the PROCEDURE. FASTQ files of studies stored on SRA (and listed through GEO) can be downloaded using the *sra toolkit* software (<http://www.ncbi.nlm.nih.gov/books/NBK158900/>). We illustrate Steps 16 - 21 with mouse training data. We also provide human pre-processed training data (see <https://github.com/pcahan1/CellNet/>).

GRN reconstruction, Step 17

CellNet is predicated upon the assumption that knowledge of GRNs that control C/T-specific expression will yield both accurate C/T classifiers and a means to prioritize perturbations to improve cell fate engineering protocols. There is a vast literature on using gene expression data to reconstruct GRNs and we refer the reader to several informative reviews^{28,29}. In the original CellNet paper, we described in detail how we used the context likelihood of relatedness (CLR) algorithm to reconstruct GRNs³⁰, and how we used the InfoMap community detection algorithm³¹ to identify sub-networks, and enrichment analysis to attribute sub-networks to specific C/Ts. We have subsequently found that we could identify C/T-specific GRNs by extracting the nodes and edges of C/T-enriched genes. We have found that the classifiers that result from using C/T GRNs generated in this way (i.e. by skipping the community detection step) performed as well as those generated using the earlier method but required less time to identify, so the most recent CellNet code uses this approach.

Train and assess CellNet, Steps 18 – 21

This section describes the steps that ultimately result in a CellNet object that can be used to analyze query expression data. The first part of this section is dedicated to assessing the performance of the C/T classifier functionality when CellNet is trained with a subset of the complete training data set, and evaluated on the remaining, independent and held out part of the training data set, resulting in one precision recall curve per C/T. This section is completed by first training CellNet on the entire training data set, then computing normalization factors on C/T GRN status metrics that will be used to scale the C/T GRN status of query samples. The result of this section is an R object that we refer to as ‘cnProc’.

Materials

Equipment

Hardware

- AMI, publicly available: CellNet ami-62065e75 (US-East N. Virginia).

CRITICAL: Pre-processing of RNA-Seq training data, and reconstructing GRNs was performed on Amazon's EC2 cloud using either c3.4×large, c3.8×large, or in the case of the human data, hs1.8×large instance types.

CRITICAL: In addition to training and assessing CellNet, and analyzing query data on AWS EC2, we have also performed the PROCEDURE on a Mac OS X (version 10.9.5), and it should also be possible to complete it using most modern incarnations of Unix-like operating systems.

CRITICAL: Our AMI has all software needed to run the PROCEDURE, but users will still have to fetch the transcriptome index and annotations that we provide and install the latest version of CellNet.

Software

- R (<http://www.r-project.org/>), version $\geq 3.2.2$.
- Optional: software such as Microsoft Excel software (commercially available from <https://products.office.com>), OpenOffice (<http://www.openoffice.org/download/>), or Google Docs for editing metadata tables
- CellNet, available from GitHub as an R package: <https://github.com/pcahan1/CellNet>
- Salmon (<http://salmon.readthedocs.io/en/latest/>), including Salmon transcriptome index file, and table that relates transcripts to canonical genes
- cutadapt (<https://cutadapt.readthedocs.io/en/stable/>)

Equipment Setup

AWS EC2 image use

The AWS EC2 image that we provide already has all of the necessary software installed. To use AWS you must first create an account at <https://console.aws.amazon.com/>. See Box 1 for a brief introduction to AWS. You can also run the protocol on your own computer or compute cluster. The software installation and setup instructions below are only necessary if you are not running the analysis on the AWS EC2 image that we provide. Box 2 describes the modifications to Steps 2, 3 and 6 that are required to run the PROCEDURE locally.

R software installation

Download and install the latest version of R from <http://cran.r-project.org/>.

Salmon installation

Download and install the latest version of Salmon from <https://github.com/COMBINE-lab/salmon/releases>. We use version 0.6.0 on our AWS image. Refer to the Salmon documentation to resolve installation issues (<http://salmon.readthedocs.io/en/latest/>). If you are using MacOS 10.12 or higher, you can use this version of Salmon: https://github.com/COMBINE-lab/salmon/files/665033/Salmon-0.7.3-pre_OSX_10.12.tar.gz.

Cutadapt installation

Cutadapt is a Python program that can be downloaded and installed on MacOS using PIP (e.g. `pip install --user --upgrade cutadapt`). For example:

```
pip install --user --upgrade cutadapt
```

The path to cutadapt needs to be added to the \$PATH environment variable.

GNU parallel installation

Use Homebrew to install GNU parallel. For example:

```
brew install parallel
```

CellNet installation

The CellNet package should be installed by executing the following R commands

```
install.packages("devtools")  
  
library("devtools")  
  
install_github("pcahan1/CellNet", ref= "rpackage")
```

Reagent Setup

Input file formats

There are three main file types used as input. First, there are comma-separated files (.csv), which can be viewed and edited using text or spreadsheet editors. Second, there are raw gene expression data FASTQ files of RNA-Seq data, which can be viewed with text editors and are pre-processed in this PROCEDURE using a combination of Salmon and custom R scripts. Third, there are R data files, which end in ".rda" and can only be accessed from within R.

Downloading example data and accessory files

The magnitude of the raw gene expression data used to train the RNA-Seq version of CellNet precludes us from making the FASTQ files for the training steps available for download. However, these files are available freely from GEO and SRA (<http://www.ncbi.nlm.nih.gov/geo/>) and the user can use the SRA accession identifiers listed under the 'sra_id' column in the file `sampTab_RS_mm_Oct_21_2016.rda` for mouse to fetch the raw data files (see below). Anticipating that some users will want to test out the protocol without the burden of fetching raw data from GEO, we provide raw expression data that can

be fetched and loaded directly into the R session starting at Step 2 and as described throughout the PROCEDURE.

Procedure

1. **Make query sample table.** Timing several minutes.

Use a spreadsheet editor to create a csv file to describe annotation information for the query data to be analyzed. Supplementary Table 1 illustrates how metadata should be formatted. This step is not necessary if you are following our example analysis.

Pre-process query data. Timing 20 minutes.

2. Getting started (see Box 1 for background information on AWS and Box 2 for a description of how perform Steps 2, 3 and 6 locally). Log in to the AWS console (<https://console.aws.amazon.com>). Search for the CellNet AMI by clicking through ‘Images’-> ‘AMIs’, then entering “CellNet” into the search field for public images. Launch the image CellNet using either c3.4×large or c3.8×large instance type. After the instance has been launched and has completed initialization, launch a terminal, and secure shell into the instance by typing the following command:

```
ssh -i aws_private_key ec2-user@instance_public_dns
```

Replace `aws_private_key` with the full path of the AWS key that you used to launch the instance. Replace `instance_public_dns` with the public DNS of your instance that can be found in the AWS console.

Launch screen by typing the following command:

```
screen
```

Launch R by typing the following command:

```
sudo R
```

Install and load the latest version of CellNet by entering the following commands:

```
library(devtools)
```

```
install_github("pcahan1/CellNet", ref= " rpackage")
```

```
q()
```

Set up disk space for the indices and fastq files by typing the following commands:

```
R
```

```
library(CellNet)
```

```
cn_setup()
```

3. Fetch transcriptome indices and annotation files using option A for mouse and B for human.
 - A. Fetch mouse transcriptome indices and annotation files
 - i. Type the following command:


```
fetch_salmon_indices(species="mouse")
```
 - B. Fetch human transcriptome indices and annotation files
 - i. Type the following command:


```
fetch_salmon_indices(species="human", iFile="salmon.index.human.050316.tgz")
```
4. Fetch and load the metadata using option A for mouse or B for human.
 - A. Fetch and load the metadata for mouse
 - i. Type the following command:


```
download.file("https://s3.amazonaws.com/CellNet/rna_seq/mouse/examples/SRP059670/st_SRP059670_example.rda",
               "st_SRP059670_example.rda")

stQuery<-
utils_loadObject("st_SRP059670_example.rda")
```
 - B. Fetch and load the metadata for human
 - i. Type the following command:


```
download.file("https://s3.amazonaws.com/CellNet/rna_seq/human/examples/SRP043684/st_SRP043684_example.rda",
               "st_SRP043684_example.rda")

stQuery<-
utils_loadObject("st_SRP043684_example.rda")
```
5. Fetch and decompress the raw query data using option A for mouse or B for human.
 - A. Fetch and decompress the raw query data for mouse
 - i. Type the following command:


```
stQuery<-cn_s3_fetchFastq("CellNet","rna_seq/mouse/examples/SRP059670",stQuery,fname="fname",
                           compressed="gz")
```
 - B. Fetch and decompress the raw query data for human
 - i. Type the following command:

```
stQuery<-cn_s3_fetchFastq("CellNet","rna_seq/human/
examples/SRP043684",stQuery,fname= "fname",
compressed= "gz")
```

?TROUBLESHOOTING

6. Estimate expression levels and save results using option A for mouse or B for human.

- A. Estimate expression levels and save results for mouse

- i. Type the following command:

```
expList<-cn_salmon(stQuery)
fname<-paste0("expList_SRP059670_example.rda")
save(expList, file=fname)
```

- B. Estimate expression levels and save results for human

- i. Type the following command:

```
expList<-cn_salmon(stQuery,salmonIndex=
"HS_GRCh38.SalmonIndex.022616",geneTabfname=
"geneToTrans_Homo_sapiens.GR
Ch38.80.exo_Jul_04_2015.R")
fname<-paste0("expList_SRP043684_example.rda")
save(expList, file=fname)
```

Analyze query data. Timing 3 minutes

- 7 Fetch and load the CellNet object that is used to analyze query data using option A for mouse or B for human

- A. Fetch and load the CellNet object that is used to analyze query data for mouse

- i. Type the following command:

```
download.file("https://s3.amazonaws.com/CellNet/
rna_seq/mouse/cnProc_MM_RS_Oct_24_2016.rda",
dest= "./cnProc_MM_RS_Oct_24_2016.rda")
cnProc<-
utils_loadObject("cnProc_MM_RS_Oct_24_2016.rda")
```

- B. Fetch and load the CellNet object that is used to analyze query data for human

- i. Type the following command:

```
download.file("https://s3.amazonaws.com/CellNet/
rna_seq/human/cnProc_RS_hs_Oct_25_2016.rda", dest=
"./cnProc_RS_hs_Oct_25_2016.rda")
```

- ```
cnProc<-
utils_loadObject("cnProc_RS_hs_Oct_25_2016.rda")
```
- 8 Apply CellNet to query data and save results.

```
cnRes1<-cn_apply(expList[['normalized']], stQuery, cnProc)
fname<-paste0("cnRes_example.rda")
save(cnRes1, file=fname)
```
  - 9 Plot C/T classification results.

```
pdf(file='hmclass_example.pdf', width=7, height=5)
cn_HmClass(cnRes1)
dev.off()
```

?TROUBLESHOOTING
  - 10 Plot GRN status using option A for mouse or B for human.
    - A. Plot GRN status for mouse
      - i. Type the following commands:

```
fname<- 'grnstats_fibroblast_example.pdf'
bOrder<-c("fibroblast_train", unique(as.vector(stQuery
$description1)), "esc_train")
cn_barplot_grnSing(cnRes1,cnProc, "fibroblast",
c("fibroblast", "esc"), bOrder, sidCol= "sra_id")
ggplot2::ggsave(fname, width=5.5, height=5)
dev.off()
fname<- 'grnstats_esc_example.pdf'
bOrder<-c("fibroblast_train", unique(as.vector(stQuery
$description1)), "esc_train")
cn_barplot_grnSing(cnRes1,cnProc, "esc",
c("fibroblast","esc"), bOrder, sidCol= "sra_id")
ggplot2::ggsave(fname, width=5.5, height=5)
dev.off()
```
    - B. Plot GRN status for human
      - i. Type the following commands:

```
fname<- 'grnstats_esc_subset_SRP043684.pdf'
bOrder<-c("esc_train", unique(as.vector(stQuery
$description2)), "neuron_train")
```

```

cn_barplot_grnSing(cnRes1,cnProc, "esc", c("esc",
"neuron"), bOrder, sidCol= "sra_id", dlevel=
"description2")

ggplot2::ggsave(fname, width=5.5, height=5)

dev.off()

fname<-'grnstats_neuron_subset_SRP043684.pdf'

bOrder<-c("esc_train", unique(as.vector(stQuery
$description2)), "neuron_train")

cn_barplot_grnSing(cnRes1,cnProc, "neuron", c("esc",
"neuron"), bOrder, sidCol= "sra_id",
dlevel='description2')

ggplot2::ggsave(fname, width=5.5, height=5)

dev.off()

```

11 Compute Network Influence Scores using option A for mouse or B for human.

A. Compute Network Influence Scores for mouse -

- i.** Compute NIS of the ESC GRN transcriptional regulators based on the day 0 samples by typing the following command:

```

rownames(stQuery)<-as.vector(stQuery$sra_id)

tfScores<-cn_nis_all(cnRes1, cnProc, "esc")

fname<-'nis_esc_example_Day0.pdf'

plot_nis(tfScores, "esc", stQuery, "Day0", dLevel=
"description1", limitTo=0)

ggplot2::ggsave(fname, width=4, height=12)

dev.off()

```

B. Compute Network Influence Scores for human.

- i.** Compute NIS of the neuron GRN transcriptional regulators based on the control iPSC-neurons by typing the following command:

```

rownames(stQuery)<-as.vector(stQuery$sra_id)

tfScores<-cn_nis_all(cnRes1, cnProc, "neuron")

fname='nis_neuron_subset_example_ctrIpsNeurons.pdf'

plot_nis(tfScores, "neuron", stQuery, "Control iPS
neurons", dLevel= "description2", limitTo=0)

ggplot2::ggsave(fname, width=4, height=12)

```

```
dev.off()
```

- 12** Fetch results. From the terminal in your computer, use scp as follows to copy the cnRes, cnProc, and expList R objects from the instance before shutting it down.

```
scp -i aws_private_key ec2-user@instance_public_dns:/media/ephemeral0/analysis/*.pdf/
```

```
scp -i aws_private_key ec2-user@instance_public_dns:/media/ephemeral0/analysis/*.rda/
```

Figures 3-5 depict the graphical outputs of these analysis steps.

Pre-process training data. Timing days

- 13** Create training metadata table by repeating Step, 1 except that the sample information corresponds to samples that the user wishes to use to train a new CellNet object. We provide a query metadata table that the user can download (see Step 16).

- 14** Repeat Step 2 to set up the EC2 instance

- 15** Fetch and load the metadata:

```
download.file("https://s3.amazonaws.com/CellNet/rna_seq/mouse/sampTab_RS_mm_Oct_21_2016.rda",
"sampTab_RS_mm_Oct_21_2016.rda")
```

```
stAll<-utils_loadObject("sampTab_RS_mm_Oct_21_2016.rda")
```

- 16** Fetch, decompress, and load the pre-processed training data, using the following command as illustrated for mouse training data::

```
download.file("https://s3.amazonaws.com/CellNet/rna_seq/mouse/expNormTrain_RS_MM_Oct_21_2016.rda",
"expNormTrain_RS_MM_Oct_21_2016.rda")
```

```
expAll<-utils_loadObject("expNormTrain_RS_MM_Oct_21_2016.rda")
```

CRITICAL STEP: You will need to modify this step if you want use your own data to train a new CellNet. First, you will need to upload FASTQ files to the instance, which can be achieved using scp. Second, you will need to estimate gene expression levels as in Step 6 using the cn_salmon function but substituting the query metadata with the training data set metadata. We recommend applying cn_salmon to subsets (rows) of the training data table distributed across several nodes.

?TROUBLESHOOTING.

Reconstruct GRNs. Timing 1-2 hours

- 17** Reconstruct C/T specific GRNs:.

```
grnProp<-cn_make_grn(stAll, expAll, species='Mm', tfs=mmTFs)
```

?TROUBLESHOOTING.

Train and assess CellNet. Timing 20-30 minutes

- 18** Split processed data into independent training and validation sets, and assess resulting classifiers:.

```
mydate<-utils_myDate()

classifierPerformance<-cn_splitMakeAssess(stAll, expAll, grnProp,
dLevelStudy='study_id',

dLevelSID= "sra_id")

fname<-paste0("classifierPerformance_", mydate, ".pdf")

pdf(file=fname, width=10, height=10)

plot_class_PRs(classifierPerformance$PRs)

dev.off()
```

An example with a description of how to interpret precision recall curves is presented in Figure 6.

?TROUBLESHOOTING

- 19** Generate the CellNet object and save it for future use:

```
cnProc<-cn_make_processor(expAll, stAll, grnProp, sidCol='sra_id')

fname<-paste0("cnProc_MM_RS_", mydate, ".rda")

save(cnProc, file=fname)
```

- 20** Examine expression of selected genes. Sometimes it may be useful to get an idea of the expression levels of a particular gene in each of the cell/tissue types of the training data. Running the following line will produce a 'rainbow' scatter plot that displays the expression levels of a specified gene in grouped by C/T:

```
library(ggplot2)

mp_rainbowPlot(cnProc$expTrain, cnProc$stTrain, "Nkx2-5",
"description1")

ggsave(file=paste("rainbowPlot_Nkx2-5_",mydate, ".pdf", sep=""),
width=4, height=3.5)

mp_rainbowPlot(cnProc$expTrain, cnProc$stTrain, "Sox2", "description1")

ggsave(file=paste("rainbowPlot_Sox2_",mydate, ".pdf",sep=""),width=4,
height=3.5)
```

Figure 7 provides example rainbow plots.

- 21** Fetch results. There are several ways to fetch the resulting files from the cloud. Below are the commands to fetch the CellNet analysis object and figures using scp. Replace `aws_private_key` with the AWS key you used to launch the

instance, and public DNS with the instance's public name, which can be found in the AWS console:

```
scp -i aws_private_key -r ec2-user@instance_public_dns:/media/ephemeral0/analysis/*.pdf./
scp -i aws_private_key ec2-user@instance_public_dns:/media/ephemeral0/analysis/*.rda./
```

Troubleshooting

Troubleshooting advice is found in Table 1.

Anticipated Results

Here we briefly describe the outputs produced in following the PROCEDURE as applied to the example data. First, Figure 3 depicts a classification heatmap in which each column represents a single input profile and each row represents one of the C/T classifiers. The intensity of the colors reflects the likelihood that the given input sample is indistinguishable from each C/T with regards to the expression of genes identified as integral the C/T GRN. In this example, there is a gradual progression from fibroblast classification to ESC classification.

Second, the panels in Figure 4A depict the extent to which the fibroblast GRN and the ESC GRN are established in each of the mouse query samples, and Figure 4B depicts the extent to which the ESC GRN and the neuron GRN are established in each of the human query samples. In some cases, we have found this GRN metric to be more sensitive than the Random Forest classifier, and so it is informative when a fate engineering attempt is not close to the target C/T yet it is on the right trajectory.

Third, Figure 5A depicts the Network Influence Score for regulators of the mouse ESC GRN relative to how these networks are configured in ESCs. Figure 5A tells us that the ESC-associated transcriptional regulators Pou5f1 (a.k.a. Oct4) and Trim28 are not as highly expressed in pre-treatment fibroblasts as in ESCs, and that their predicted targets are dysregulated. Therefore, we would predict that the up-regulation of these transcription factors would improve the ESC GRN status. Likewise, Figure 5B depicts the NIS for regulators of the human neuron GRN relative to how these networks are configured in neurons. Figure 5B tells us that the neuron-associated transcriptional regulators MYT1L and SNCA are not as highly expressed in iPSC-derived neurons as in neurons from the training data, and that their predicted targets are dysregulated.

Supplementary Material

Refer to Web version on PubMed Central for supplementary material.

Acknowledgments

PC is supported by NIDDK (K01DK096013). We thank Evan Appleton for helpful comments on the protocol.

References

1. Murry CE, Keller G. Differentiation of Embryonic Stem Cells to Clinically Relevant Populations: Lessons from Embryonic Development. *Cell*. 2008; 132:661–680. [PubMed: 18295582]
2. Ieda M, et al. Direct Reprogramming of Fibroblasts into Functional Cardiomyocytes by Defined Factors. *Cell*. 2010; 142:375–386. [PubMed: 20691899]
3. Kyba M, Perlingeiro RCR, Daley GQ. HoxB4 confers definitive lymphoid-myeloid engraftment potential on embryonic stem cell and yolk sac hematopoietic progenitors. *Cell*. 2002; 109:29–37. [PubMed: 11955444]
4. Bock C, et al. Reference Maps of Human ES and iPS Cell Variation Enable High-Throughput Characterization of Pluripotent Cell Lines. *Cell*. 2011; 144:439–452. [PubMed: 21295703]
5. McKinney-Freeman S, et al. The Transcriptional Landscape of Hematopoietic Stem Cell Ontogeny. *Cell Stem Cell*. 2012; 11:701–714. [PubMed: 23122293]
6. Hussein SMI, et al. Genome-wide characterization of the routes to pluripotency. *Nature*. 2015; 516:198–206.
7. Davidson EH, Erwin DH. Gene regulatory networks and the evolution of animal body plans. *Science*. 2006; 311:796–800. [PubMed: 16469913]
8. Cahan P, et al. CellNet: Network Biology Applied to Stem Cell Engineering. *Cell*. 2014; 158:903–915. [PubMed: 25126793]
9. Morris SA, et al. Dissecting Engineered Cell Types and Enhancing Cell Fate Conversion via CellNet. *Cell*. 2014; 158:889–902. [PubMed: 25126792]
10. Berger DR, Ware BR, Davidson MD, Allsup SR, Khetani SR. Enhancing the functional maturity of iPSC-derived human hepatocytes via controlled presentation of cell-cell interactions in vitro. *Hepatology*. 2014; doi: 10.1002/hep.27621
11. Godoy P, et al. Gene networks and transcription factor motifs defining the differentiation of stem cells into hepatocyte-like cells. *J Hepatol*. 2015; 63:934–942. [PubMed: 26022688]
12. Song G, et al. Direct Reprogramming of Hepatic Myofibroblasts into Hepatocytes In Vivo Attenuates Liver Fibrosis. *Cell Stem Cell*. 2016; 18:797–808. [PubMed: 26923201]
13. Cao N, et al. Conversion of human fibroblasts into functional cardiomyocytes by small molecules. *Science*. 2016; 352:1216–1220. [PubMed: 27127239]
14. Uosaki H, et al. Transcriptional Landscape of Cardiomyocyte Maturation. *Cell Rep*. 2015; 13:1705–1716. [PubMed: 26586429]
15. Lu YF, et al. Engineered Murine HSCs Reconstitute Multi-lineage Hematopoiesis and Adaptive Immunity. *Cell Rep*. 2016; 17:3178–3192. [PubMed: 28009288]
16. Pavlidis N, Fizazi K. Carcinoma of unknown primary (CUP). *Crit Rev Oncol Hematol*. 2009; 69:271–278. [PubMed: 18977667]
17. Bian Q, Cahan P. Computational Tools for Stem Cell Biology. *Trends in Biotechnology*. 2016; 34:993–1009. [PubMed: 27318512]
18. Muller FJ, Loring JF. Network biology: A compass for stem-cell differentiation. *Nature*. 2014; 513:498–499. [PubMed: 25254472]
19. Avior Y, Biancotti JC, Benvenisty N. TeratoScore: Assessing the Differentiation Potential of Human Pluripotent Stem Cells by Quantitative Expression Analysis of Teratomas. *Stem Cell Reports*. 2015; 4:967–974. [PubMed: 26070610]
20. Roost MS, et al. KeyGenes, a Tool to Probe Tissue Differentiation Using a Human Fetal Transcriptional Atlas. *Stem Cell Reports*. 2015; 4:1112–1124. [PubMed: 26028532]
21. Rackham OJL, et al. A predictive computational framework for direct reprogramming between human cell types. *Nat Genet*. 2016; 48:331–335. [PubMed: 26780608]
22. D'Alessio AC, et al. A Systematic Approach to Identify Candidate Transcription Factors that Control Cell Identity. *Stem Cell Reports*. 2015; doi: 10.1016/j.stemcr.2015.09.016
23. Cieply B, et al. Multiphasic and Dynamic Changes in Alternative Splicing during Induction of Pluripotency Are Coordinated by Numerous RNA-Binding Proteins. *Cell Rep*. 2016; 15:247–255. [PubMed: 27050523]

24. Mertens J, et al. Differential responses to lithium in hyperexcitable neurons from patients with bipolar disorder. *Nature*. 2015; 527:95–99. [PubMed: 26524527]
25. Martin M. Cutadapt removes adapter sequences from high-throughput sequencing reads. *EMBnet journal*. 2011; 17:10–12.
26. Patro, R., Duggal, G., Love, MI., Irizarry, RA., Kingsford, C. bioRxiv. Cold Spring Harbor Labs Journals; 2015. Salmon provides accurate, fast, and bias-aware transcript expression estimates using dual-phase inference. 021592
27. Anders S, Huber W. Differential expression analysis for sequence count data. *Genome Biol*. 2010; 11:R106. [PubMed: 20979621]
28. Margolin AA, et al. Reverse engineering cellular networks. *Nat Protoc*. 2006; 1:662–671. [PubMed: 17406294]
29. Margolin AA, Califano A. Theory and limitations of genetic network inference from microarray data. *Annals of the New York Academy of Sciences*. 2007; 1115:51–72. [PubMed: 17925348]
30. Faith JJ, et al. Large-Scale Mapping and Validation of Escherichia coli Transcriptional Regulation from a Compendium of Expression Profiles. *PLoS Biol*. 2007; 5:e8. [PubMed: 17214507]
31. Rosvall M, Bergstrom CT. Maps of information flow reveal community structure in complex networks. 2007
32. Mouse ENCODE Consortium et al. An encyclopedia of mouse DNA elements (Mouse ENCODE). *Genome Biol*. 2012; 13:418. [PubMed: 22889292]
33. Xu H, et al. ESCAPE: database for integrating high-content published data collected from human and mouse embryonic stem cells. *Database (Oxford)*. 2013; 2013:bat045. [PubMed: 23794736]
34. Correa-Cerro LS, et al. Generation of mouse ES cell lines engineered for the forced induction of transcription factors. *Sci Rep*. 2011; 1

Box 1**Amazon Web Services (AWS)**

AWS is a suite of cloud based resources that includes data storage (using S3) and computation (using EC2). AWS provides access to high performance computing resources without the overhead of setting up and maintain compute clusters locally. One of the main benefits is that you can configure virtual computers, termed images or Amazon machine images (AMIs), to have specific software and libraries pre-installed, which makes standardized and reproducible analysis feasible. We have configured an EC2 image so that this PROCEDURE can be reproduced by you. To access the image, you need to sign up for AWS at <https://aws.amazon.com/console/>. Instructions for getting started with AWS can be found at <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/get-set-up-for-amazon-ec2.html>. These instructions include the creation of a key pair file required to launch and securely access an instance. We refer to this key pair file as the AWS key throughout the PROCEDURE. AWS charges users based on increments of services used. For example, the c4.8xlarge instance currently costs \$1.591/hour. Once you have launched an instance, you will be able to login into to it using ssh and your key pair. Likewise, you will use scp with your key pair to push files to and fetch files from your instance.

Box 2**Modifications to Steps 2, 3 and 6 of the Procedure to pre-process data locally**

Some changes need to be made so that you can run CellNet and Salmon locally.

Local Steps 2 and 3

- 2 Install and load the latest version of CellNet by entering the following commands after launching R:

```
library(devtools)

install_github("pcahan1/CellNet", ref= "rpackage")

q()
```

Set up disk space for the indices and fastq files, and set salmonVersion to the version of Salmon that you have installed locally by typing the following commands:.

```
R

library(CellNet)

cn_setup(local=TRUE)
```

- 3 Fetch transcriptome indices and annotation files using option A for mouse or option B for human:

- A. Fetch mouse transcriptome indices and annotation files

- i. Type the following command:

```
iFileMouse<- "salmon.index.mouse.122116.tgz"

fetchIndexHandler(destination= "ref/", species=
"mouse", iFile=iFileMouse)
```

- B. Fetch human transcriptome indices and annotation files.

- i. Type the following command:

```
iFileHuman<- "salmon.index.human.122116.tgz"

fetchIndexHandler(destination= "ref/", species=
"human", iFile=iFileHuman)
```

Local Step 6]

- 6 Firstly replace "path/to/Salmon/bin/" below with the path to where you installed Salmon:

```
iFileMouse<- "salmon.index.mouse.122116"

iFileHuman<- "salmon.index.human.122116"

pathToSalmon<- "path/to/Salmon/bin/"
```

Then, estimate expression levels and save results using option A for mouse or B for human.

A. Estimate expression levels and save results for mouse

i. Use the following command:

```
expList<-cn_salmon(stQuery, refDir=
"ref",salmonIndex=iFileMouse,
salmonPath=pathToSalmon)

fname<-paste0("expList_SRP059670_example.rda")

save(expList, file=fname)
```

B. Estimate expression levels and save results for human

i. Use the following command:

```
expList<-cn_salmon(stQuery,refDir= "ref",
salmonIndex=iFileHuman,geneTabfname=
"geneToTrans_Homo_sapiens.GRCh38.80.exo_Jul_0
4_201 5.R",salmonPath=pathToSalmon)

fname<-paste0("expList_SRP043684_example.rda")

save(expList, file=fname)
```

Figure 1. Inputs and outputs of CellNet

CellNet takes as input gene expression data from cell fate engineering experiments and returns three outputs as described in the text. Previously CellNet was applied to microarray data but here we describe how to use RNA-Seq data.

Figure 2.

Outline of the Procedure. The overall PROCEDURE, indicating the steps to which each section corresponds, the required equipment, inputs, outputs, expected run time, and our recommendations for whether to execute each step locally or on the cloud. We have provided intermediate results of each section so that the user can begin the PROCEDURE at any point. If the user is only analyzing human or mouse query data, then they only need to follow Steps 1-12. If the user wants to add new cell or tissue types or to create a CellNet platform for a different species, then Steps 13 - 21 should be followed to train and assess CellNet.

Figure 3. Classification heatmap of the example query data. Columns represent query samples. Rows represent C/Ts of the training data. Each square is colored by the classification score of the query sample for each C/T. Scores range from 0 (i.e. distinct from the C/T of the training data) to 1 (i.e. indistinguishable from the C/T of the training data). (A) Mouse example data. (B) Human example data.

Figure 4. C/T specific GRN status of fibroblasts as they are reprogrammed to pluripotency
GRN status indicates the extent to which a C/T GRN is established in the training (dark blue) and query (light blue) samples. The raw GRN status is computed as the mean z-score of all genes in a C/T GRN weighted by their importance to the associated C/T classifier. The raw GRN status is then normalized to the mean raw GRN status of the training data samples of the given C/T (see ref. 8). Error bars represent mean $\pm$ one standard deviation. Number of replicates per group varies from 1 (for Prox1 GFP plus and minus in panel B) to 182 (for fibroblast_train in panel A). (A) Mouse example data. Left is the fibroblast GRN status and right panel is the ESC GRN status. (B) Human example data. Left panel is the ESC GRN status and right panel is the neuron GRN status.

Figure 5.

Network influence score (NIS). The transcriptional regulators of the C/T GRN are shown on the y axis with the NIS on the x-axis. The NIS prioritizes TFs such that their experimental perturbation is predicted to improve the target C/T classification. The NIS of a TF is computed based on three components (see ref. 8). The first component is the extent to which the TF is dysregulated as compared to its expected value in the target C/T. The second component is the number of predicted targets of the TF. And the third component is the extent to which the target genes are dysregulated. (A) NIS of ESC TFs in the starting fibroblast population of the mouse example data. (B) NIS of neuron TFs in the control (non-disease) iPSC-derived neurons of the human example data. The circular data points are outliers defined as those that have values exceeding 1.5 times the extremes of the inter-quartile range.

Figure 6.

Precision recall curves of each murine RNA-Seq C/T classifier. The x-axis is the sensitivity, or the proportion of samples that are from the given C/T and are classified as such. Recall is equivalent to sensitivity. The y-axis is the precision, defined as the proportion of samples classified as the given C/T that are truly derived from that C/T. Each point represents the precision versus sensitivity at a given classification score threshold. As the threshold is increased, the recall tends to decrease as the precision increases.

Figure 7. Scatter plots showing the expression of Nkx2-5 (A) and Sox2 (B) across the murine RNA-Seq training data sets. Each point represents the expression of the gene in a single training data set. Different colors represent different C/Ts. Y-axis represents the different C/Ts. X-axis represents the expression level.

Table 1
Troubleshooting advice

Step	Problem	Possible reason	Possible solution
9	Low classification score for query samples	Target C/T is not in the training data	Add target C/T to training data set and re-make cnProc
5; 16	Processing raw FASTQ files fails	Insufficient drive space	Increase disk space or process samples in smaller increments
17	GRN reconstruction fails	Insufficient RAM	Execute on larger instance type
18	Classifier assessment is poor	Incorrect training sample annotation or poor training data quality	Double check sample annotation and perform quality control on reads to ensure good mapping rates to target transcriptome