1 Introduction

One of the central cellular processes underlying development is transcriptional regulation. During development, changes in transcription factor activity induce chromatin modifications, chromatin remodelling and ultimately a differential recruitment of the basal transcriptional machinery[1]. Modelling the dynamics of gene regulation is therefore essential to better understand why a cellular dynamic processes progresses through several steps, and what goes wrong in the case of disease.

The dynamics of gene regulation has classically been studied using time series data[2]. When dynamic processes progress asynchronously, such as in hematopoiesis, time series data are usually obtained by sorting different transition states and assessing bulk gene expression and transcription factor binding within the population[3, 4, 5, 6]. Alternatively, time series data can also be generated by synchronizing the dynamic process between cells. However, issues with time-resolution, heterogeneity and good in vivo synchronization models can often limit the predictive power of the dynamic models of gene regulation which can be constructed[2].

One of the main advantages of single-cell transcriptomics is the ability to quantify the exact cellular state of thousands of cells per experiment. The intercellular heterogeneity caused by naturally occurring biological stochasticity [7] can be exploited to predict regulatory interactions between transcription factors (TFs) and their target genes. The computational tools that infer gene regulatory networks (GRNs) from omics datasets are called network inference (NI) methods.

Several studies have highlighted how some regulatory interactions can be very dynamic while others show evidence of being static during consecutive developmental stages[8, 9]. Since regulatory interactions are context-dependent[10], attempting to create an accurate model of those processes by inferring a static regulatory network may have limited relevance. Case-wise NI methods[1] avoid predicting a static GRN and instead infer one GRN per cell (or per sample, for bulk omics data). The case-wise GRNs – or 'case-wise regulomes' – are analysed similarly to single-cell transcriptomics data; for example by clustering, inferring trajectories, or finding differentially activated interactions.

In order to compute a case-wise GRN for a single sample, Kuijjer et al.[11] and Liu et al.[12] employ similar strategies, namely by computing the difference of computing a static GRN for all the cases, and computing a static GRN for all the cases minus one. Since this procedure needs to be repeated for every case in the dataset, and because NI methods are already amongst the most computationally intensive analyses to perform on omics data, this methodology is not applicable for large omics datasets. Another case-wise NI method, SCENIC[13] infers case-wise GRNs by first inferring a static GRN using GENIE3[14]. GENIE3 is a static NI method which uses Random Forests[15] feature importance scores to prioritise candidate regulators for a particular target gene. SCENIC then post-processes the static GRN to determine whether an interaction is enriched for particular cases, resulting in a case-wise GRN. In

[1]Case-wise NI is sometimes also called sample-specific NI or case-specific NI.

short, while several case-wise NI methods thus already exist, their implementation consisted of post-processing a static GRN to arrive at a case-wise GRN.

In this work, we introduce `bred`, the first 'true' case-wise NI method. `bred` uses case-wise feature importance[16] score rather than the regular Random Forest feature importance scores. In addition to predicting regulatory strength of an interaction, `bred` also predicts the regulatory effect of an interaction – that is, whether a regulator activates or represses the target. We demonstrate `bred` by applying it on a single-cell dataset of 25'000 hematopoietic cells from the Tabula Muris project[17], and to a collection of 15'000 bulk omics samples from The Cancer Genome Atlas project[18].

2 Results

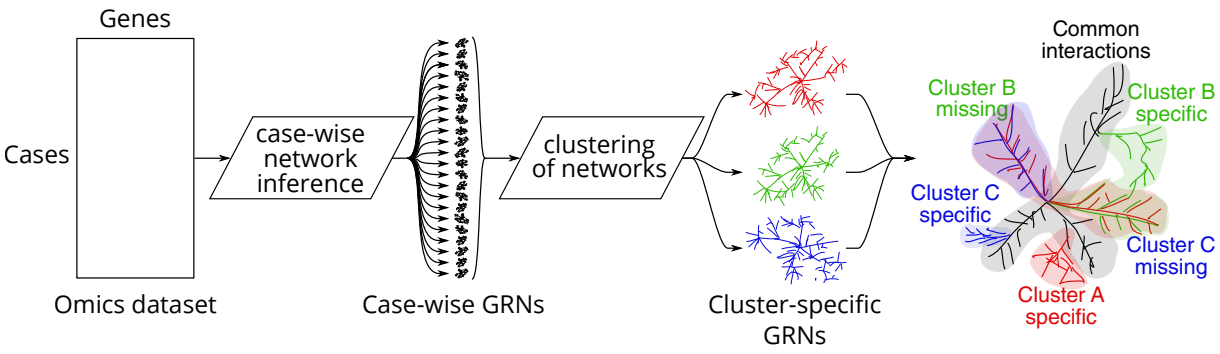

Figure 1: A schematic overview of the workings of the `bred` algorithm.

3 Discussion

4 Methods

4.1 Building the case-wise GRNs

Having to infer a static GRN (Figure 4A) can be reduced to a simpler problem, namely for every target T, predict which of the potential regulators regulate T (Figure 4B). This simplification allowed GENIE3[14] to use Random Forest's[15] feature importance scores for inferring GRNs. Namely, a Random Forest is trained to predict the expression of a target gene of interest from the expression of potential regulators. The resulting Random Forest inherently allows to extract a feature importance score by observing the effect of each regulator in making a good prediction for the target expression. As in GENIE3, the target expression is first scaled to normalise feature importance scores across different targets.

We make the same simplification in order to build case-wise GRNs, also using Random Forests to compute the feature importance scores. A Random Forest consists of K trees, each of which produces feature importance scores, and the feature importance scores of a forest is simply the mean feature importance scores of each of the trees.

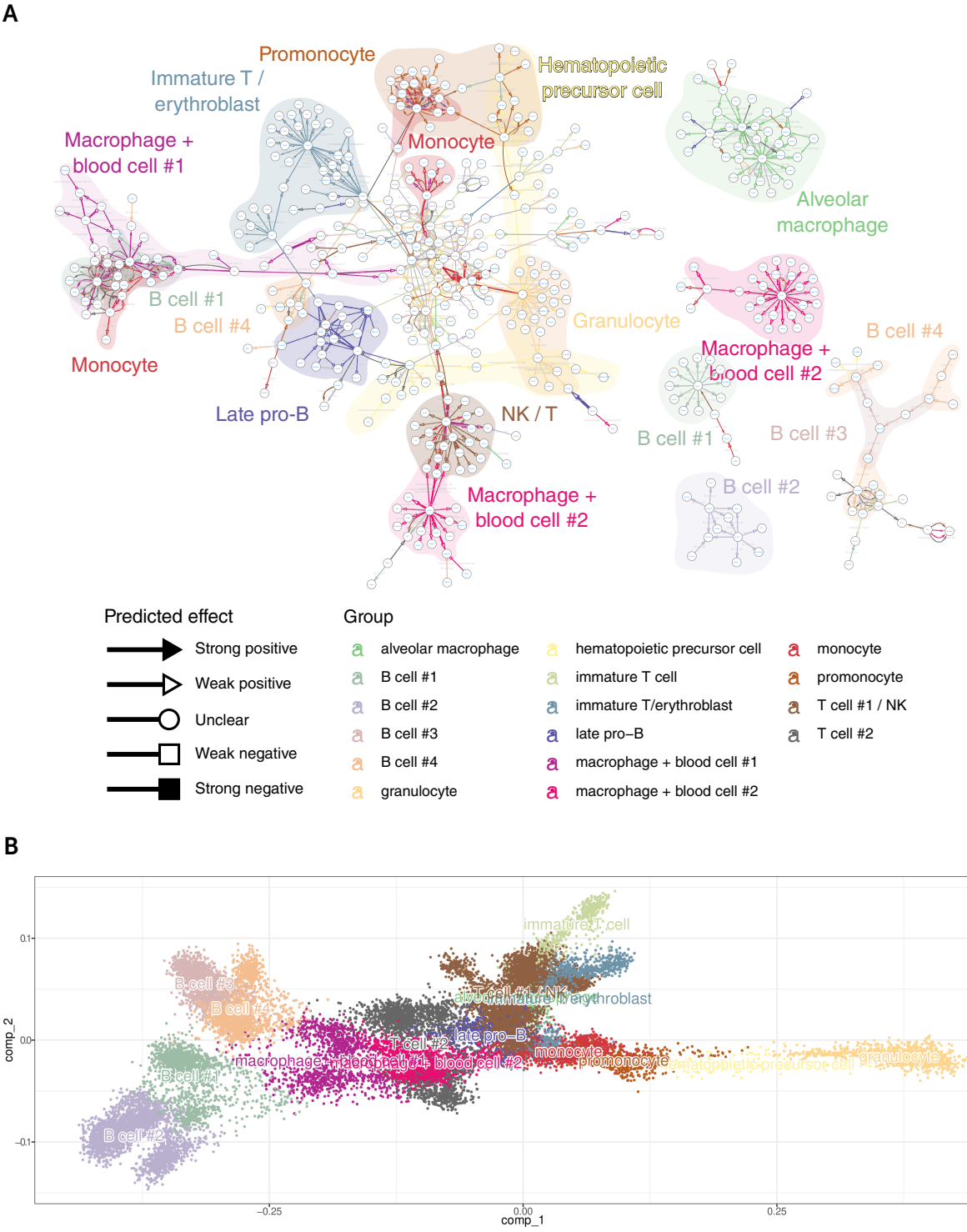

Figure 2: Hematopoietic cells from the Tabula Muris project.

Computing the case-wise feature importances of a tree consists of the following 8 steps (Figure 5). The 'randomness' of a Random Forest is due to only using a subset of the samples in the dataset in order to build a single decision tree. The samples are split into two groups, the 'in-bag' data and the 'out-of-bag' data (Figure 5A). A decision tree[19] is trained on the in-bag expression of the potential regulators in trying to predict the in-bag target gene expression (Figure 5B). The target expression of the out-of-bag samples is predicted using the

decision tree (Figure 5C), and the squared error between the real and target expression is computed (Figure 5D). For each sample in the out-of-bag set, this vector represents how well the decision tree was able to predict the expression of the target gene.

The next few steps are repeated for every potential regulator R_i. Within the out-of-bag samples, the expression of R_i is randomly shuffled. The target expression of the out-of-bag samples is again calculated (Figure 5F), as well as the squared error between the real target expression and the predicted expression is calculated (Figure 5G). The importance of regulator R_i for an out-of-bag sample S_j is defined as the increase in squared error between the predicted target expression and the real target expression, after perturbing the expression of R_i (Figure 5H).

Steps F-G are repeated for every potential regulator R_i. By aggregating all of the feature importance scores over all the samples, regulators and targets, we obtain an M-by-N-by-P tensor[2].

A moderately-sized dataset could contain $M = 10'000$ samples, $N = 2'000$ regulators, and $P = 10'000$ target genes. Due to memory constrains, only interactions with an average importance value (across all samples) higher than a minimum threshold are retained.

To compute the case-wise GRNs, we implemented the abovementioned methodology in C++ in a modified version of the `ranger` R/C++ package[20].

4.2 Predicting the effect of an interaction

To predict the effect of a potential regulator R_i on a target gene T for a given tree, the Pearson correlation is calculated between the difference in regulator expression (before and after shuffling the values), and the difference in target expression prediction.

$$\begin{aligned}\text{effect}(R_i \to T) &= \text{cor}(x, y),\\ &\text{with } x = \text{expr\_shuffled}[:, R_i] - \text{expr}[:, R_i],\\ &\text{and } y = \text{predict}(\text{tree}, \text{expr\_shuffled}) - \text{predict}(\text{tree}, \text{expr}).\end{aligned}$$

The Pearson correlation between two variables x and y is usually defined as shown in Equation 1. Computing r_{xy} for each (regulator, target) pairs, across all trees, would require storing large amounts of data.

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x}) \times (y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2} \times \sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2}} \tag{1}$$

However, by rearranging the formula, it can be defined as Equation 2.

$$r_{xy} = \frac{\sum(x_i \times y_i) - \sum x \times \sum y / n}{\sqrt{(\sum x_i^2 - (\sum x)^2/n)} \times \sqrt{(\sum y_i^2 - (\sum y)^2/n)}} \tag{2}$$

[2]This is the origin of the name of the method, "`bred`".

For every regulator R_i during a perturbation in a given tree, only 6 values need to be stored, namely $A = \sum x_i$, $B = \sum y_i$, $C = n$, $D = \sum x_i \times y_i$, $E = \sum x_i \times x_i$, and $F = \sum y_i \times y_i$.

For every (regulator, target) pair, these values are summed, and the r_{xy} is calculated as shown in Equation 3.

$$r_{xy} = \frac{D - A \times B/C}{\sqrt{(E - A^2/C)} \times \sqrt{(F - B^2/C)}} \tag{3}$$

The following cutoffs were used to determine the effect.

- Strong negative: $r_{xy} < -0.4$
- Weak negative: $-0.4 \leq r_{xy} < -0.2$
- Unclear: $-0.2 \leq r_{xy} \leq 0.2$
- Weak positive: $0.2 < r_{xy} \leq 0.4$
- Strong positive: $0.4 < r_{xy}$

4.3 Clustering of case-wise GRNs

To perform downstream analysis on the cases, first a k-nearest neighbour (kNN) graph of the cases is computed. In order for the kNN graph to better emphasise similarities in GRNs rather than absolute euclidean distances, we first reduce the dimensionality of the case-by-interaction matrix to case-by-20 matrix using Landmark Multi-Dimensional Scaling[21] with a Spearman rank distance metric.

Next, KD-trees are used to calculate the kNN graph efficiently. The cases in the dataset are visualised and clusted using the Fruchterman-Reingold[22] and Louvain clustering[23], respectively.

The following R packages provided implementations for each of these algorithms: lmds, RANN, igraph[24].

4.4 Data from the Tabula Muris project

4.5 Data from The Canger Genome Atlas project

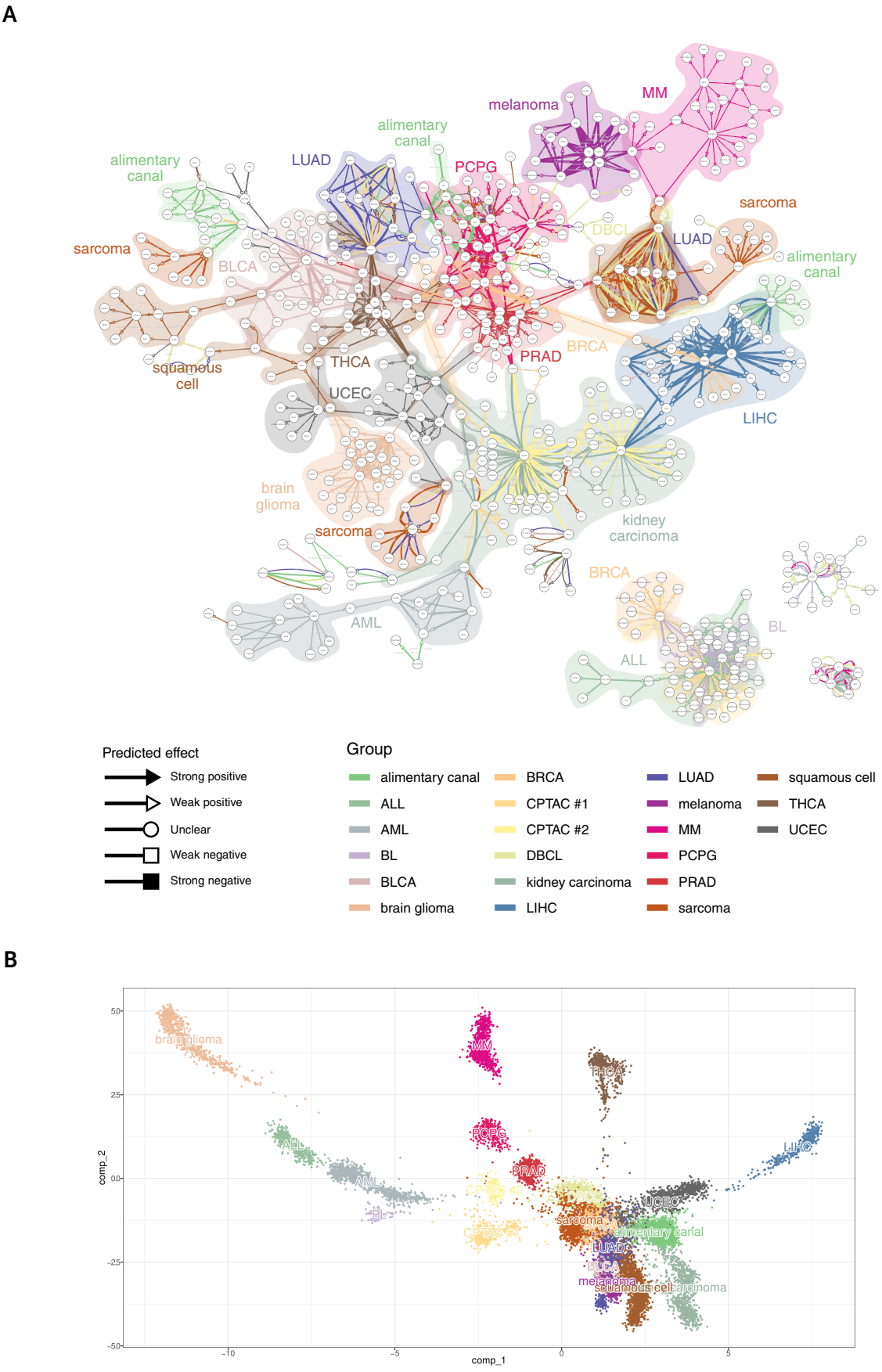

Figure 3: The Cancer Genome Atlas.

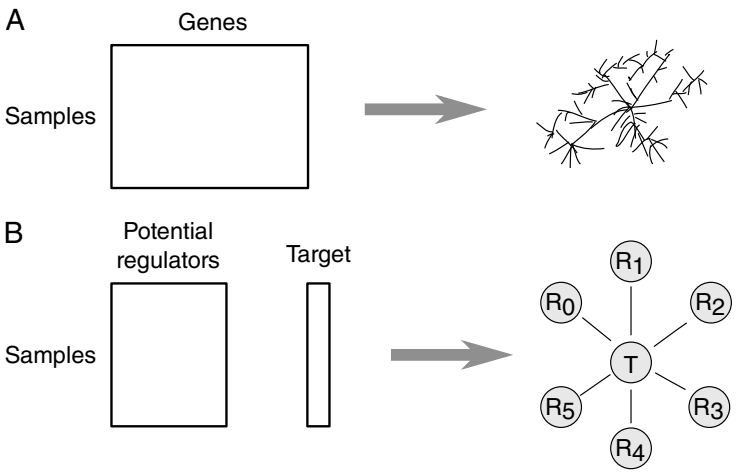

Figure 4: A: Inferring a gene regulatory network from an omics dataset can be reduced to a simpler problem. **B:** Given the expression of a target of interest and a set of potential regulators, predict which regulators regulate the target gene.

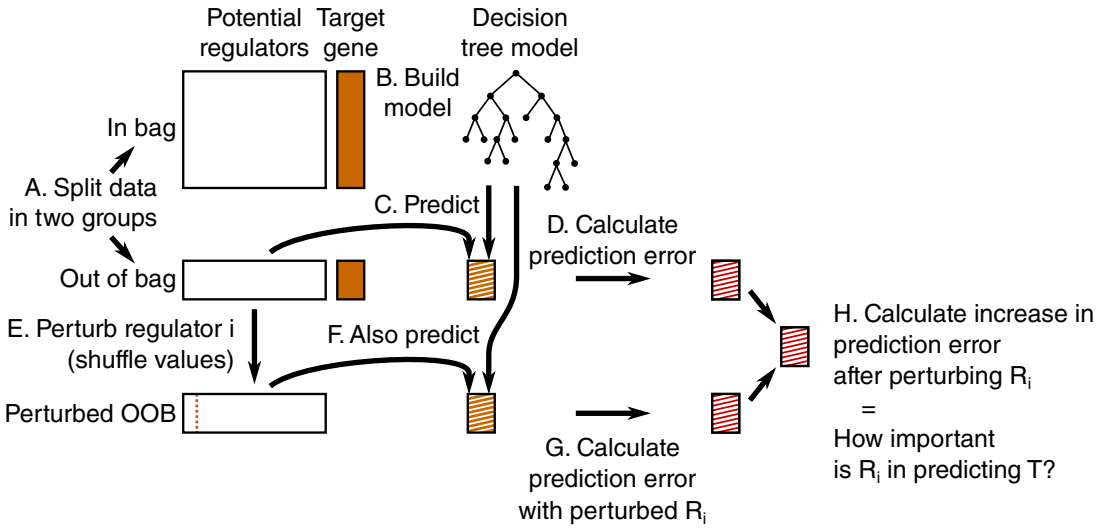

Figure 5: Calculating the feature importance score for one decision tree and one target consists of 8 distinct steps. A: Randomly split the data into two groups, the in-bag data and the out-of-bag data. **B:** The in-bag data is used to train a decision tree to try to predict the expression of the target gene from the expression values of the regulators. **C:** The decision tree is used to predict the gene expression of the target gene of the out-of-bag samples. **D:** Sample-specific squared error values are computed. **E:** Repeat steps E-H for every regulator R_i. Perturb the expression of regulator R_i in the out-of-bag samples. **F:** Again predict the gene expression of the target gene with the perturbed expression values. **G:** Again compute the sample-specific squared error values. **H:** The difference between the prediction error on the perturbed dataset versus the prediction error on the unperturbed is the importance in R_i in predicting T

5 References

[1] Antoine Coulon et al. "Eukaryotic Transcriptional Dynamics: From Single Molecules to Cell Populations". In: *Nature Reviews Genetics* 14 (July 9, 2013), p. 572. DOI: 10.1038/nrg3484.

[2] Ziv Bar-Joseph, Anthony Gitter, and Itamar Simon. "Studying and Modelling Dynamic Biological Processes Using Time-Series Gene Expression Data". In: *Nat. Rev. Genet.* 13.8 (Aug. 2012), pp. 552–564.

[3] Noa Novershtern et al. "Densely Interconnected Transcriptional Circuits Control Cell States in Human Hematopoiesis". In: *Cell* 144.2 (2011), pp. 296–309.

[4] Gillian May et al. "Dynamic Analysis of Gene Expression and Genome-Wide Transcription Factor Binding during Lineage Specification of Multipotent Progenitors". In: *Cell Stem Cell* 13.6 (2013), pp. 754–768.

[5] Vladimir Jojic et al. "Identification of Transcriptional Regulators in the Mouse Immune System". In: *Nat. Immunol.* 14.6 (2013), pp. 633–643. DOI: 10.1038/ni.2587.Identification.

[6] Debbie K Goode et al. "Dynamic Gene Regulatory Networks Drive Hematopoietic Specification and Differentiation". In: *Dev. Cell* 36.5 (2016), pp. 572–587.

[7] Olivia Padovan-Merhar and Arjun Raj. "Using Variability in Gene Expression as a Tool for Studying Gene Regulation". In: *Wiley Interdisciplinary Reviews. Systems Biology and Medicine* 5.6 (Nov. 2013), pp. 751–759. ISSN: 1939-005X. DOI: 10.1002/wsbm.1243. pmid: 23996796.

[8] Victoria Moignard et al. "Characterization of Transcriptional Networks in Blood Stem and Progenitor Cells Using High-Throughput Single-Cell Gene Expression Analysis". In: *Nat. Cell Biol.* 15.4 (Apr. 2013), pp. 363–372.

[9] Cristina Pina et al. "Single-Cell Network Analysis Identifies DDIT3 as a Nodal Lineage Regulator in Hematopoiesis." In: *Cell reports* 11.10 (2015), pp. 1503–1510. ISSN: 2211-1247. DOI: 10.1016/j.celrep.2015.05.016. pmid: 26051941.

[10] Balázs Papp and Stephen Oliver. "Genome-Wide Analysis of the Context-Dependence of Regulatory Networks". In: *Genome Biology* 6.2 (Jan. 27, 2005), p. 206. ISSN: 1474-760X. DOI: 10.1186/gb-2005-6-2-206.

[11] Marieke Lydia Kuijjer et al. "Estimating Sample-Specific Regulatory Networks". In: *iScience* 14 (Mar. 28, 2019), pp. 226–240. ISSN: 2589-0042. DOI: 10.1016/j.isci.2019.03.021. pmid: 30981959.

[12] Xiaoping Liu et al. "Personalized Characterization of Diseases Using Sample-Specific Networks". In: *Nucleic Acids Research* 44.22 (2016), e164–e164. ISSN: 0305-1048. DOI: 10.1093/nar/gkw772. pmid: 27596597.

[13] Sara Aibar et al. "SCENIC: Single-Cell Regulatory Network Inference and Clustering". In: *Nature Methods* (Oct. 2017). ISSN: 1548-7091. DOI: 10.1038/nmeth.4463.

[14] Vân Anh Huynh-Thu et al. "Inferring Regulatory Networks from Expression Data Using Tree-Based Methods". In: *PLoS ONE* 5.9 (Jan. 2010), e12776. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0012776. pmid: 20927193.

[15] Leo Breiman. "Random Forests". In: *Machine Learning* 45 (2001), pp. 5–32.

[16] Ruo Xu, Dan Nettleton, and Daniel J. Nordman. "Case-Specific Random Forests". In: *Journal of Computational and Graphical Statistics* 25.1 (2016), pp. 49–65. ISSN: 1061-8600. DOI: 10.1080/10618600.2014.983641.

[17] Nicholas Schaum et al. "Single-Cell Transcriptomics of 20 Mouse Organs Creates a Tabula Muris". In: *Nature* 562.7727 (Oct. 2018), pp. 367–372. ISSN: 1476-4687. DOI: 10.1038/s41586-018-0590-4.

[18] John N Weinstein et al. "The Cancer Genome Atlas Pan-Cancer Analysis Project." In: *Nature genetics* 45.10 (Oct. 2013), pp. 1113–20. ISSN: 1546-1718. DOI: 10.1038/ng.2764. pmid: 24071849.

[19] L Breiman et al. *Classification and Regression Trees*. Wadsworth Publishing Company, 1984.

[20] Marvin N Wright and Andreas Ziegler. "Ranger: A Fast Implementation of Random Forests for High Dimensional Data in C++ and R". In: *Journal of Statistical Software* 77.1 (Mar. 2017). DOI: 10.18637/jss.v077.i01.

[21] Seunghak Lee and Seungjin Choi. "Landmark MDS Ensemble". In: *Pattern Recognition* 42.9 (Sept. 2009), pp. 2045–2053. ISSN: 00313203. DOI: 10.1016/j.patcog.2008.11.039.

[22] Thomas M. J. Fruchterman and Edward M. Reingold. "Graph Drawing by Force-Directed Placement". In: *Software: Practice and Experience* 21.11 (1991), pp. 1129–1164. ISSN: 1097-024X. DOI: 10.1002/spe.4380211102.

[23] Vincent D Blondel et al. "Fast Unfolding of Communities in Large Networks". In: *Journal of Statistical Mechanics: Theory and Experiment* 2008.10 (Oct. 9, 2008), P10008. ISSN: 1742-5468. DOI: 10.1088/1742-5468/2008/10/p10008.

[24] Gabor Csardi and Tamas Nepusz. "The Igraph Software Package for Complex Network Research". In: *InterJournal, Complex Systems* 1695.5 (2006), pp. 1–9.