Using deep learning to model the hierarchical structure and function of a cell

Jianzhu Ma[1,5], Michael Ku Yu[1,2,5], Samson Fong[1,3,5], Keiichiro Ono[1], Eric Sage[1], Barry Demchak[1], Roded Sharan[4] & Trey Ideker[1–3]

Although artificial neural networks are powerful classifiers, their internal structures are hard to interpret. In the life sciences, extensive knowledge of cell biology provides an opportunity to design visible neural networks (VNNs) that couple the model's inner workings to those of real systems. Here we develop DCell, a VNN embedded in the hierarchical structure of 2,526 subsystems comprising a eukaryotic cell (http://d-cell.ucsd.edu/). Trained on several million genotypes, DCell simulates cellular growth nearly as accurately as laboratory observations. During simulation, genotypes induce patterns of subsystem activities, enabling *in silico* investigations of the molecular mechanisms underlying genotype–phenotype associations. These mechanisms can be validated, and many are unexpected; some are governed by Boolean logic. Cumulatively, 80% of the importance for growth prediction is captured by 484 subsystems (21%), reflecting the emergence of a complex phenotype. DCell provides a foundation for decoding the genetics of disease, drug resistance and synthetic life.

Deep learning has revolutionized the field of artificial intelligence by enabling machines to perform human activities like seeing, listening and speaking[1–6]. Such systems are constructed from many-layered, 'deep', artificial neural networks (ANNs) inspired by actual neural networks in the brain and how they process patterns. The function of the ANN is created during a training phase, in which the model learns to capture as accurately as possible the correct answer, or output, that should be returned for each example input pattern. In this way, machine vision learns to recognize objects like dogs, people and faces, and machine players learn to distinguish good from bad moves in games like chess and Go[7].

In modern ANN architectures, the connections between neurons as well as their strengths are subject to extensive mathematical optimization, leading to densely entangled network structures that are neither tied to an actual physical system nor based on human reasoning. Consequently, it is typically difficult to grasp how any particular set of neurons relates to system function. For instance, AlphaGo beats top human players[7], but examination of its underlying network yields little insight into the rules behind its moves or how these are encoded by neurons. These are so-called black boxes[8], in which the input–output function accurately models an actual system, but the internal structure does not (**Fig. 1a**). Such models, while undoubtedly useful, are insufficient in cases where simulation is needed not only of system function but also of system structure. In particular, many applications in biology and medicine seek to model both functional outcome and the mechanisms leading to that outcome so that these can be understood and manipulated through drugs, genes or environment.

Here we report DCell, an interpretable or 'visible' neural network (VNN) simulating a basic eukaryotic cell. The structure of this model is formulated from extensive prior knowledge of the cell's hierarchy of subsystems documented for the budding yeast *Saccharomyces cerevisiae*. This knowledge is drawn from either of two sources: the Gene Ontology (GO), a literature-curated reference database from which we extracted 2,526 cellular subsystems (intracellular components, processes or functions)[9]; or the Clique-eXtracted Ontology (CliXO), an alternative ontology of similar size inferred from large-scale molecular data sets rather than literature curation[10,11]. While CliXO and GO overlap in 38% of subsystems, some in CliXO are apparent in large-scale data sets but not yet characterized in literature, whereas some in GO are documented in the literature but difficult to identify in big data. Subsystems in these ontologies are interrelated through hierarchical parent–child relationships of membership or containment. Such hierarchies form a natural bridge from variations in genotype, at the scale of nucleotides and genes, to variations in phenotype, at the scale of cells and organisms[12,13].

The function of DCell is learned during a training phase, in which perturbations to genes propagate through the hierarchy to impact parent subsystems that contain them, giving rise to functional changes in protein complexes, biological processes, organelles and, ultimately, a predicted response at the level of a cell

[1]Department of Medicine, University of California San Diego, La Jolla, California , USA. [2]Program in Bioinformatics, University of California San Diego, La Jolla, California, USA. [3]Department of Bioengineering, University of California San Diego, La Jolla, California, USA. [4]Blavatnik School of Computer Science, Tel Aviv University, Tel Aviv, Israel. [5]These authors contributed equally to this work. Correspondence should be addressed to T.I. (tideker@ucsd.edu).

RECEIVED 19 JULY 2017; ACCEPTED 7 FEBRUARY 2018; PUBLISHED ONLINE 5 MARCH 2018; DOI:10.1038/NMETH.4627

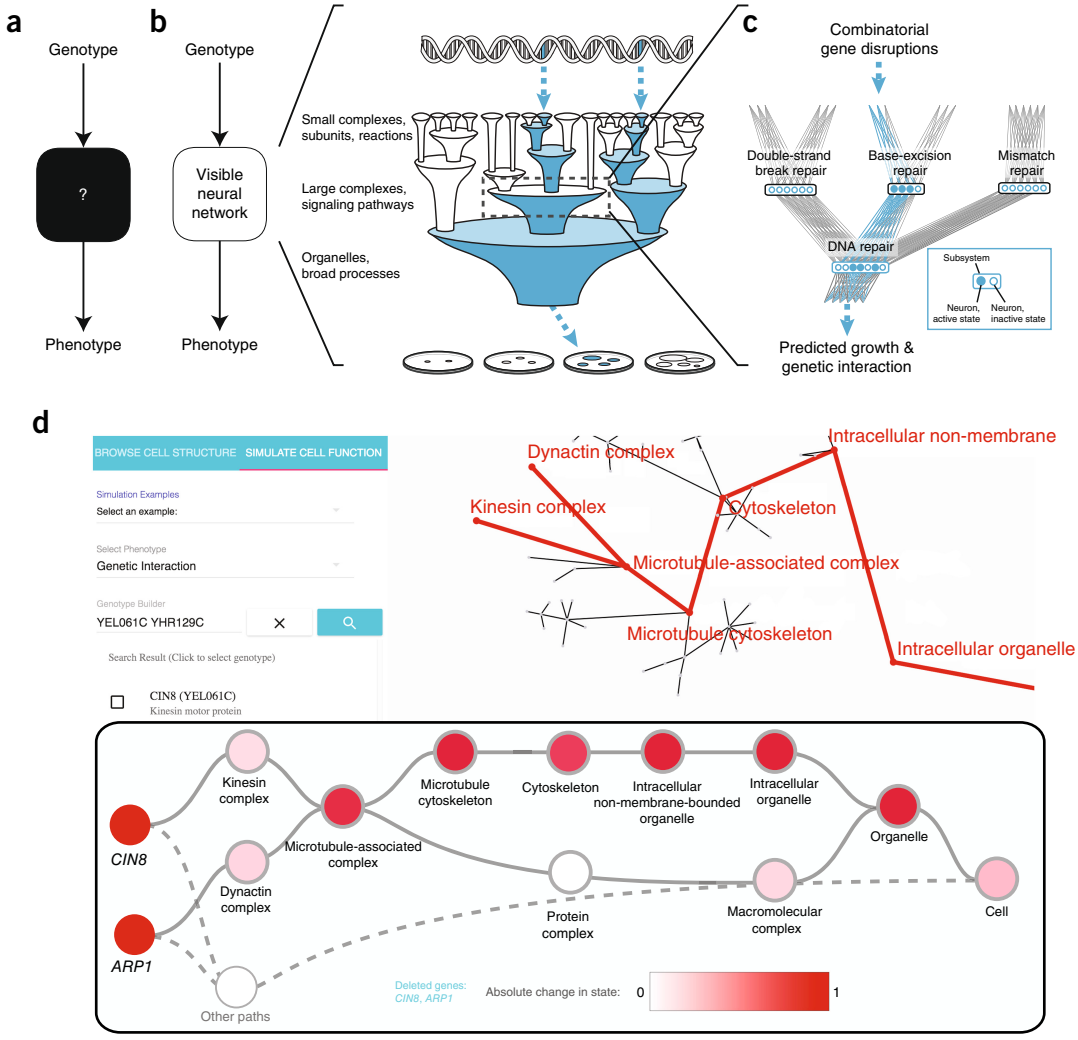

Figure 1 | Modeling system structure and function with visible learning. (**a**) A conventional neural network translates input to output as a black box without knowledge of system structure. (**b**) In a visible neural network, input–output translation is based on prior knowledge. In DCell, gene-disruption genotypes (top) are translated to cell-growth predictions (bottom) through a hierarchy of cell subsystems (middle). (**c**) A neural network is embedded in the prior structure using multiple neurons per subsystem. (**d**) Screen capture of DCell online service.

growth phenotype (**Fig. 1b**). Previously, we saw that hierarchical groups of genes in an ontology could be used to formulate input features for such phenotypic predictions[12,13]. However, these features were provided to standard black-box machine-learning models, which could not be interpreted biologically. Here, we embed the structure of the deep neural network directly in the biological hierarchy, enabling transparent biological interpretation.

RESULTS

DCell design

In DCell, the functional state of each subsystem is represented by a bank of neurons (**Fig. 1c**). Connectivity of these neurons is set to mirror the biological hierarchy, so that they take inputs only from neurons of child subsystems and send outputs only to neurons of parent subsystems, with weights determined during training. The use of multiple neurons (ranging from 20 to 1,075 per system; see Online Methods) acknowledges that cellular components are often multifunctional, with states that are too complex to be captured by a single neuron[14]. The input layer of the hierarchy comprises the genes, whereas the output layer, or root, is a single neuron representing cell phenotype. By this design, the VNN embedded in GO includes 97,181 neurons; the corresponding model for CliXO includes 22,167 neurons. The depth of both networks is 12 layers, on par with deep neural networks in other fields[7].

Performance in genotype–phenotype translation

Given this architecture, we taught DCell to predict phenotypes related to cellular fitness, a model genotype-to-phenotype translation task (see Online Methods). Extensive training was made possible by an existing compendium of yeast growth phenotypes measured for single- and double-gene-deletion genotypes, comprising several million genotype–phenotype training examples[15,16]. We considered two related phenotypes: (i) capacity for growth measured by colony size relative to wild-type cells, and (ii) for double gene deletions, genetic interaction score measured as the difference in colony size from that expected from the corresponding single-gene deletions. Predicting genetic interaction

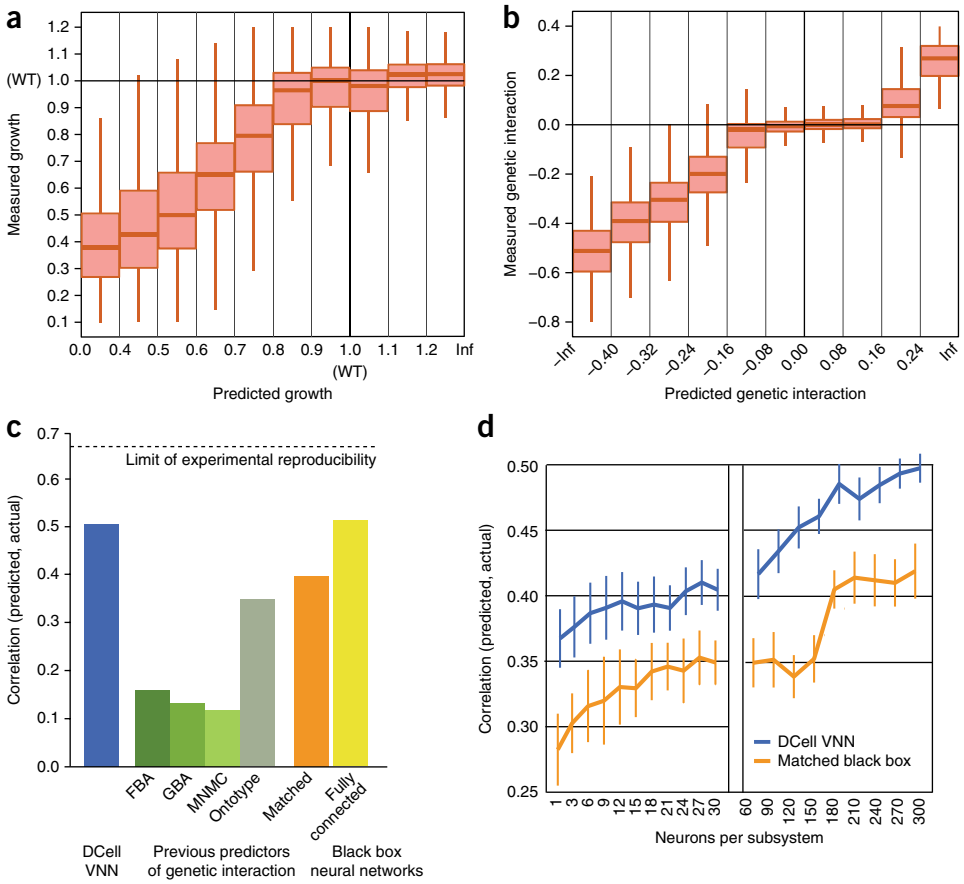

Figure 2 | Prediction of cell viability and genetic interaction phenotypes. (**a**) Measured versus predicted cell viability relative to wild type (WT = 1) on the Costanzo *et al.*[16] data set. (**b**) Measured versus predicted genetic interaction scores for each double-gene-disruption genotype; genetic interactions between the disrupted genes can be positive (epistasis), zero (noninteraction), or negative (synthetic sickness or lethality). (**c**) Model performance expressed as the correlation between measured and predicted genetic interaction scores. Performance of DCell (blue) is compared to that of previous methods for predicting genetic interactions (green): FBA, Flux Balance Analysis[17]; GBA, Guilt By Association[18]; MNMC, Multi-Network Multi-Classifier[19]; and Ontotype[13]. Performance is also shown for matched black-box structures in which gene-to-subsystem mappings are randomly permuted (orange, average of ten randomizations) or for fully connected neural networks with the same number of layers as DCell (yellow). Correlations were calculated across gene pairs that met an interaction significance criterion of $P < 0.05$. DCell based on GO hierarchy; for DCell based on CliXO, see **Supplementary Figure 1**. (**d**) Predictive performance versus number of neurons per subsystem. Performance measure and two structural hierarchies as in **c**.

represents a harder task than predicting absolute growth, as it requires learning of nonlinear effects beyond superposition of single-gene-deletion genotypes. Based on the training examples, the weights of input connections to each neuron were optimized by stochastic gradient descent computed by backpropagation. For execution and inspection of this DCell model, we created an interactive website at http://d-cell.ucsd.edu/ (**Fig. 1d**).

We found that DCell could make accurate phenotypic predictions for both growth (**Fig. 2a**) and genetic interaction (**Fig. 2b**). It outperformed previous predictors, including those based on metabolic models[17] and protein–protein interaction networks[18,19], as well as a hierarchical method not related to deep learning (**Fig. 2c** and **Supplementary Fig. 1**)[13]. We also compared performance to black-box ANNs of several types. First, we constructed ANNs with matching structure to DCell but permuting the assignment of genes to subsystems. Predictive performance decreased substantially (**Fig. 2c**) and was restored only after the number of neurons was increased by an order of magnitude (**Fig. 2d**). Thus, the biological hierarchy provides important information not found in randomized versions. Second, we constructed a fully connected ANN with the same number of layers and neurons as DCell but unlimited connectivity between adjacent layers. Despite these extra parameters, performance of this fully connected model was not substantially better (**Fig. 2c**).

From prediction to mechanistic interpretation

Unlike standard ANNs, DCell's simulations were tied to an extensive hierarchy of internal biological subsystems with states that could be queried. This 'visible' aspect raised the possibility that, unlike black-box neural networks, DCell could be used for *in silico* studies of biological mechanism. We focus on four major types of such studies in the next sections.

Explaining a genotype–phenotype association

A fundamental goal of genetics is to explain the molecular mechanisms linking changes in genotype to changes in phenotype. To generate such explanations automatically, we used DCell to simulate the impact of a genotypic change, relative to wild type, on the states of all cellular subsystems in the model. We considered subsystems with substantial changes to be candidate explanations

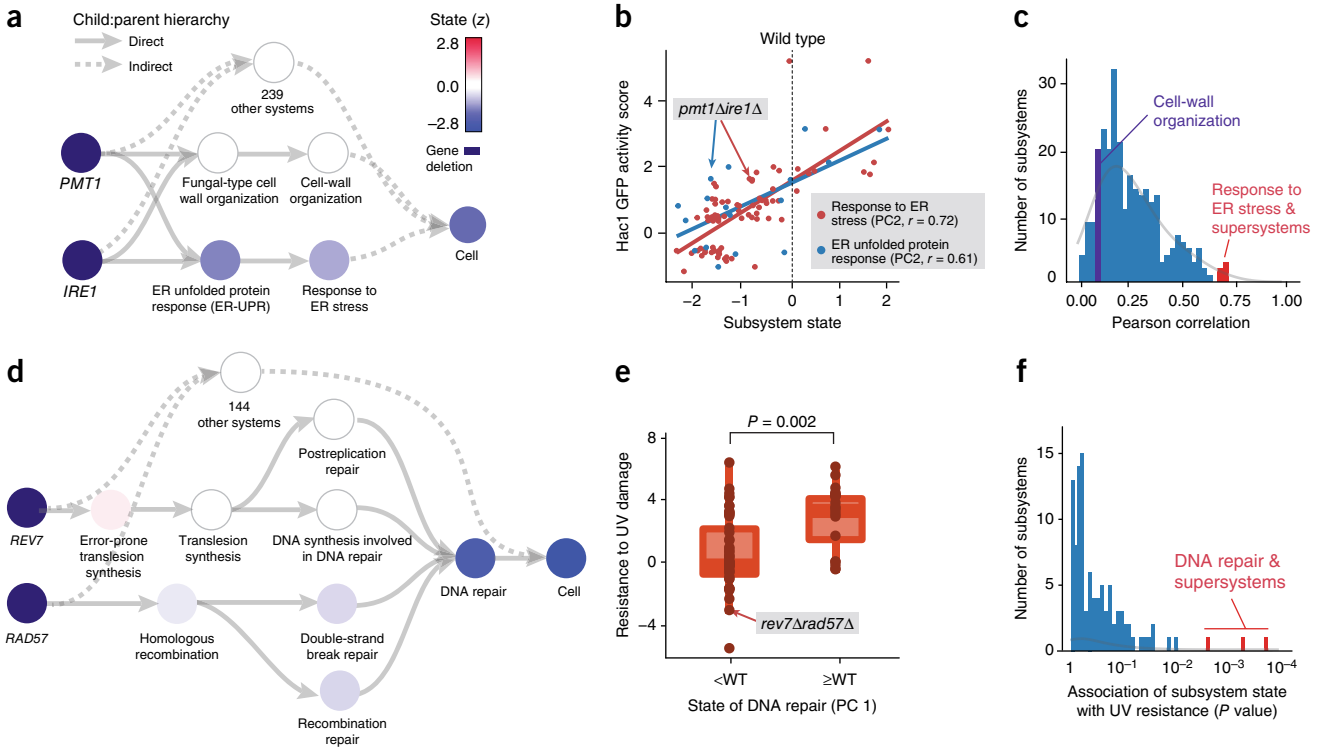

Figure 3 | Interpretation of genotype–phenotype associations. (**a**) DCell simulations generate a hierarchy of candidate subsystems that can explain the association of the *pmt1Δire1Δ* genotype with a negative genetic interaction phenotype (synthetic lethality). Subsystem states are represented by their neuron values, reduced to the first two principal components (PCs), which capture >75% of the total variance. The color of the node for each subsystem shows its PC2 expressed as a *z*-score with zero mean and unit s.d. The wild-type genotype produces $z = 0$ for all subsystems (see Online Methods). (**b**) Correspondence between Hac1 GFP activity and the functional states of ER-UPR (blue) or its parent subsystem, response to ER stress (red). Points represent genotypes, with *pmt1Δire1Δ* genotype indicated. Subsystem state is the *z*-score of PC2 as in **a**. (**c**) Distribution of correlations between Hac1 GFP activity and the states of every other subsystem containing at least ten genotypes with measured GFP activity. (**d**) DCell simulations indicate that DNA repair is a highly altered subsystem that explains the slow growth phenotype of *rev7Δrad57Δ*. The color of the node for each subsystem indicates its PC1 expressed as a z-score (see key in **a**). (**e**) Experimental resistance to UV damage plotted against the simulated state of DNA repair in DCell, separated into two classes: above and below wild-type value. Significance measured by Mann–Whitney *U* test. (**f**) Distribution of the associations between UV-damage resistance and the states of other subsystems containing at least ten genotypes with measured UV resistance. Panels **a**–**c** use genetic interaction prediction as the phenotypic readout; **d**–**f** use growth. All panels implement GO as the model of system structure.

in translation of genotype to phenotype, whereas those without state changes—typically the vast majority—were excluded from consideration. For example, to explain the severe growth defect caused by *pmt1Δire1Δ*, disrupting the genes *PMT1* and *IRE1*, we simulated this genotype with DCell and examined the 243 subsystems incorporating *PMT1* or *IRE1* at any level of the GO hierarchy (subsystems that are ancestors of one or both genes). These subsystems encompassed functions of *PMT1* or *IRE1* in the endoplasmic reticulum unfolded protein response (ER-UPR)[20,21], cell wall organization and integrity[22,23], and many other processes (**Fig. 3a**). Examining the simulated states of these candidate subsystems (values of their neurons), we found that ER-UPR output was substantially reduced compared to that in the wild type, whereas cell-wall organization and other subsystems were relatively unaffected (**Fig. 3a**).

To validate this simulated decrease, we examined a data set measuring abundance of green fluorescent protein (GFP) driven by a promoter responsive to Hac1, a key transcriptional activator of ER-UPR, over numerous pairwise gene disruptions[24]. Hac1 activity was decreased in the *pmt1Δire1Δ* genotype compared with that in the wild type, consistent with model simulations (**Fig. 3b**). Moreover, we found that the simulated state of ER-UPR was well correlated with experimental Hac1 activity, not only for this genotype but across all relevant gene disruptions in the data set (**Fig. 3b**). To address the concern that Hac1 activity might associate nonspecifically with state changes in many subsystems, not just those related to ER, we examined its correlation with the simulated states of every subsystem in DCell. High correlation was observed only for ER-UPR and supersystems (**Fig. 3c**), demonstrating specific validation. In this way, DCell was used to test among competing mechanistic hypotheses for a genotype–phenotype relationship.

In explaining genotype–phenotype associations, a key requirement is that the state of a subsystem *in silico* approximate its true state *in vivo*. To further validate this capability, we examined the subsystem of DNA repair (**Fig. 3d**) which, like ER-UPR, had been experimentally interrogated over many double gene deletions[25]. In particular, DNA repair status had been characterized by resistance to ultraviolet radiation (UV), a model DNA-damaging agent[26]. Once again we saw good agreement between model and experiment; the simulated state of DNA repair significantly tracked experimental UV resistance across genotypes (**Fig. 3e**; Mann—Whitney *U* test, $P = 0.002$), in a manner highly specific to this subsystem (**Fig. 3f**; Mann—Whitney *U* test, $P < 0.01$).

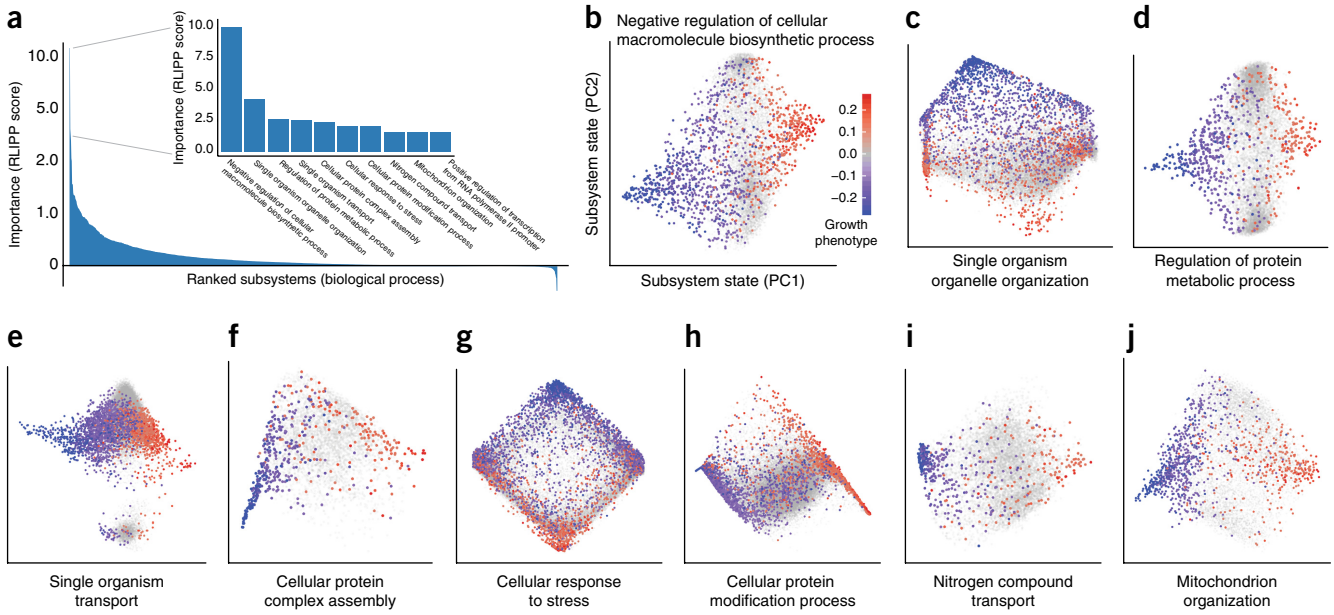

Figure 4 | Identification of subsystems important for cell growth. (**a**) Ranking of all cellular subsystems in GO (*x*-axis) by their importance in determining genetic interactions underlying growth phenotype (RLIPP score, *y*-axis). Inset, ten highest scoring subsystems. Positive RLIPP corresponds to increases in predictive power relative to children; negative RLIPP corresponds to decreases in power. (**b**–**j**) 2D state maps of informative subsystems (PC2 versus PC1). All axes are on the same scale, referring to the PC weights. **Supplementary Figure 2** provides equivalent information for the CliXO hierarchy.

Prioritizing important systems that determine a phenotype

Beyond individual explanations for genotype–phenotype associations, a critical question is whether a complex phenotype such as growth depends on equal contributions from many subsystems or is dominated by a few. To address this question, we reasoned that the overall importance of a subsystem can be computed quantitatively as the degree to which its state is more predictive of phenotype than the states of its children—a metric we called relative local improvement in predictive power (RLIPP; see Online Methods). We observed that RLIPP approximately followed a Pareto (power-law) distribution, in which a few subsystems are important for model predictions, with a long tail of less important systems (**Fig. 4a**). In particular, 80% of the cumulative importance was captured by 21% of subsystems (the Pareto 80/20 rule[27]), while >88% of subsystems retained some improvement in phenotypic prediction over their children (RLIPP > 0). The GO subsystem of greatest individual importance in cell growth was 'negative regulation of cellular macromolecule biosynthesis', which organizes cellular circuits that inhibit biosynthesis and, as evidenced by DCell simulations, can lead to strong increases in growth when disrupted. Other subsystems important for growth related to the proper function of organelles, biomolecular transport, stress response, protein modification, and assembly of complexes (**Fig. 4b**–**j**).

Characterization of genetic logic underlying a process

A third application of DCell relates to the mathematical functions by which the neurons representing each subsystem integrate information. We investigated whether these functions could be reduced to simple forms, such as Boolean logic gates, which are easily interpreted (see Online Methods). This analysis found 1,119 subsystems at least partly governed by Boolean logic (44% of GO; **Supplementary Table 1**). For instance, the state of mitochondrial respiratory chain (**Fig. 5a**), while relatively high in wild-type cells, was driven low by disruptions in any of the several enzymatic complexes involved in electron transport, such as complexes III and IV (**Fig. 5b**). Thus the mechanism underlying the functions encompassed by mitochondrial respiratory chain resembles a logical AND gate (**Fig. 5c**). We also observed many cases of OR, XOR, and (A not B), although the AND configuration arose most frequently. The remaining subsystems did not map clearly to Boolean functions, suggesting machinery that is more complex than an on–off switch.

Discovery of new biological processes and states

Finally, we investigated the extent to which DCell simulations rely on new cellular subsystems not previously reported in the biological literature. For this purpose, we examined the version of DCell for which the subsystem hierarchy had been structured from large-scale molecular data sets (CliXO) rather than previous literature (GO). This CliXO hierarchy was based on detection of nested community structures within a large gene-interaction network summarizing numerous types of experimental data, including protein–protein binding, gene coexpression, gene coevolution, genetic linkage, and so on (see Online Methods). While some of the network communities corresponded to cellular subsystems that were already well known, many others were previously undocumented, suggesting subsystems that are potentially new.

Indeed, when CliXO was used as the structure for DCell's deep neural network, we found 236 subsystems that were previously undocumented and also had high RLIPP importance scores for genotype–phenotype translation (**Supplementary Table 2**). One example was CliXO:10651, a previously undocumented process

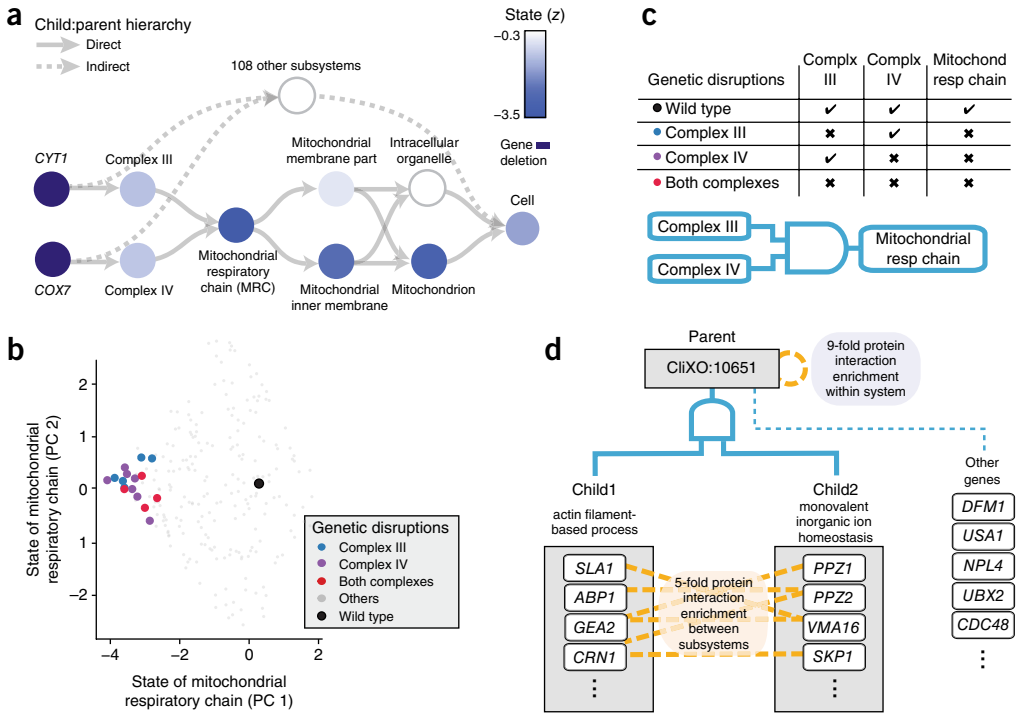

Figure 5 | Analysis of subsystem functional logic. (**a**) DCell simulations explain the effects of a *cyt1Δcox7Δ* genotype by a causal hierarchy of subsystems involving changes to the mitochondrial respiratory chain (MRC). Display similar to **Figure 3a**, with node color indicating PC1 z-score according to the key. (**b**) 2D state map of MRC plotted as in **Figure 4b**. Genotypes disrupting MRC complex III only, MRC complex IV only, or both complexes are indicated. (**c**) Truth table relating the state of MRC to the states of its children. The logic resembles an AND gate, pictured. A check represents normal wild-type output; an "x" represents decreased output. (**d**) The schematic shows a newly identified system (CliXO:10651), identified by DCell to encode a logical AND integrating the states of two well-characterized children. Names of children are cross-annotated from the corresponding enriched GO terms: Actin filament-based process, GO:0030029; Monovalent inorganic ion homeostasis, GO:0030004.

ranking among the top ten systems important for growth prediction. We found that CliXO had inferred this system based on the elevated density of protein–protein interactions observed among its 154 genes (**Fig. 5d**, nine-fold enrichment, $P < 10^{-200}$). These interactions interconnected two subsystems that are individually much better understood, relating to actin filaments and ion homeostasis (five-fold enrichment between subsystems, $P = 0.00029$). The simulated state of CliXO:10651 was governed approximately by a Boolean AND of the states of its two subsystems, both being required to maintain wild-type status. These findings are supported by previous reports that homeostasis of ions, such as iron, regulates the level of oxidative stress, which in turn disrupts actin cytoskeletal organization[28,29].

As a second example we considered CliXO:10582, a novel subsystem of 71 genes (**Fig. 6a**). Although many of these genes had known roles in DNA repair, this grouping had not been previously recognized. Examination of the hierarchical model structure revealed that CliXO:10582 interconnects components of three known DNA repair subsystems, postreplication repair, mismatch repair, and nonrecombinational repair, based on a very high density of protein–protein interactions among these components (**Fig. 6a**). Revisiting the experimental data on resistance to UV-induced DNA damage[25] (**Fig. 3e,f**), we saw that the simulated state of CliXO:10582 strongly correlated with experimental UV resistance across genotypes (**Fig. 6b**). This association was stronger than for any child and, in fact, for any other CliXO subsystem interrogated by the experimental data (**Fig. 6c**). Mathematically, the state of CliXO:10582 was not well captured by Boolean logic but by a weighted linear summation of the states of the three child systems, with postreplication repair having the greatest single contribution (**Fig. 6d**). Thus, DCell has identified a novel organization of subsystems which specifically coordinate the response to UV damage. For the eight genes in this system not previously known to function in DNA repair (green nodes, **Fig. 6a**), the evidence summarized by the model—that these gene products physically interact within a larger cluster of known DNA repair factors, and that they functionally manifest with the same UV sensitivity phenotype when disrupted—creates a compelling case for further studies.

DISCUSSION

A direct route to interpretable neural networks is to encode not only function but form. Here, we have explored such visible learning in the context of cell biology, by incorporating an unprecedented collection of knowledge[10,11,30] and data[15,16,31] to simultaneously simulate cell hierarchical structure and function. DCell captured nearly all phenotypic variation in cellular growth, a classic complex phenotype, including much of the less understood nonadditive portion due to genetic interactions (**Fig. 2a–c**). Armed with this explanatory power, the model simulated the intermediate functional states of thousands of cellular subsystems. Knowledge of these states enables *in silico* studies of molecular mechanism, including dissection of subsystems important to growth phenotype, identification of new subsystems, and

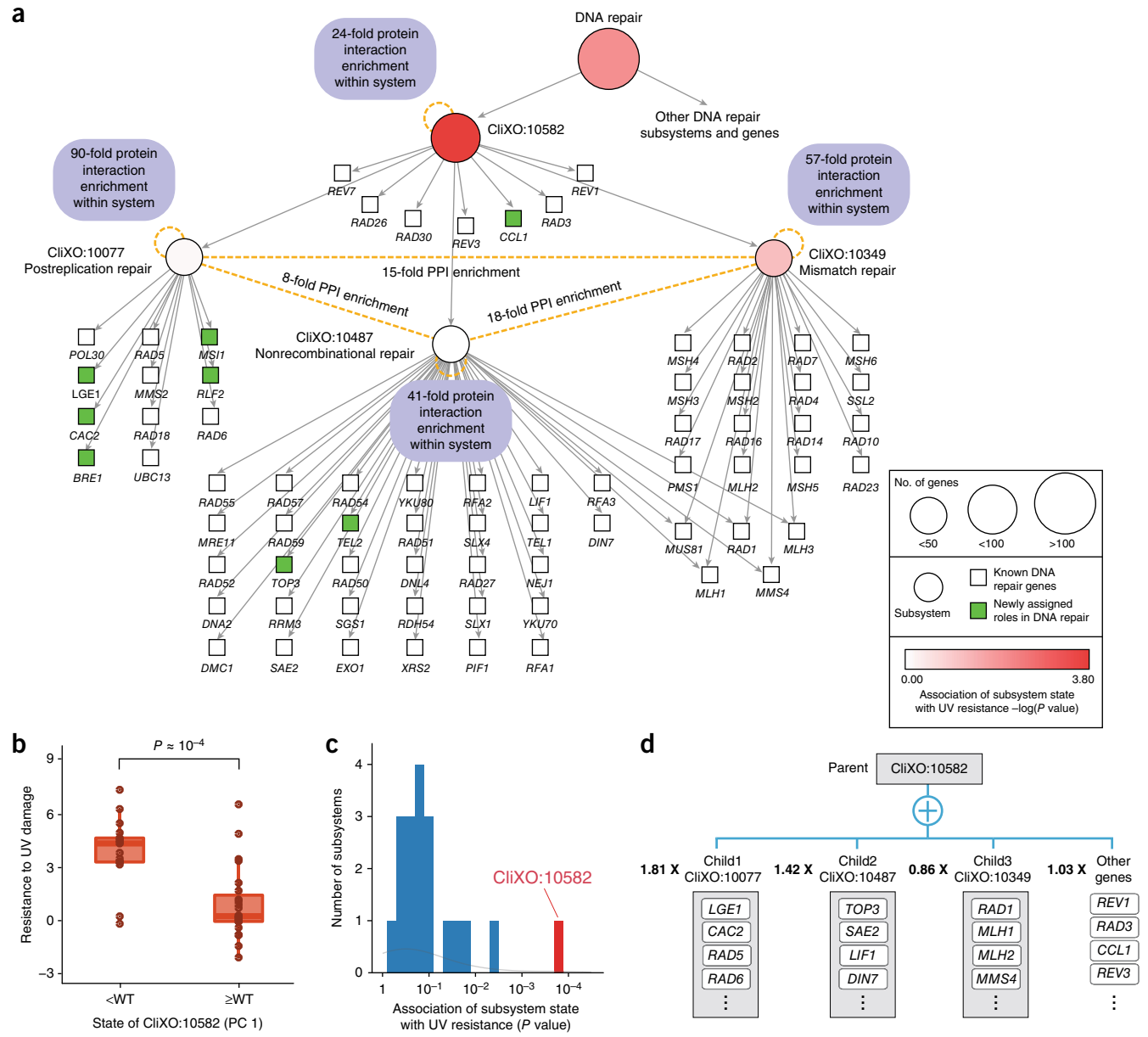

Figure 6 | Analysis of a new DNA repair subsystem. (**a**) A new hierarchical organization of DNA repair. Subsystems and their hierarchical relations are identified by CliXO, while the states of subsystems are inferred by simulation of DCell embedded in this structure. (**b**) Experimental resistance to UV damage plotted against the state of CliXO:10582, separated into two classes: above and below wild-type value. Significance measured by Mann–Whitney *U* test. (**c**) Distribution of associations between UV resistance and the states of all CliXO subsystems with at least ten genotypes with measured UV resistance. (**d**) Weighted linear summation approximating the state of CliXO:10582 as a function of the states of its children. Numbers in bold are weights. Subsystem states are the PC1s of their neurons.

reduction of subsystem functions, where possible, to Boolean logic (**Figs. 3–6**).

Methodologically, our approach works toward a synthesis of statistical genetics and systems biology. State-of-the-art methods in statistical genetics[32,33] are based on linear regression of phenotype against the independent effects of genetic polymorphisms, without modeling the underlying molecular mechanisms that give rise to nonlinearity and genetic interaction. Separately, studies in systems biology capture molecular mechanisms using mathematical models[34,35], but such models typically do not have the breadth for large-scale genetic dissection of phenotype. DCell bridges these two avenues—its neural network encodes a complex nonlinear regression, an extension of statistical genetics, in which the additional complexity is enabled by a hierarchical mechanistic model, an extension of systems biology. In contrast to other mechanistic models that have attempted large-scale genotype–phenotype prediction[13,36], the framework of hierarchical neural networks is very general and expressive, such that a large class of biological structures and functions can be represented. For example, our earlier approach[13] used hierarchical knowledge of subsystems to create new features based on the number of gene disruptions in a subsystem, but these features were predetermined

before modeling, and thus nothing was learned about the real functions encoded by subsystems.

It is also instructive to view DCell in the context of previous research in interpretable machine learning, in which the notion of interpretability has been defined in different ways[37]. One direction has been to perform a *post hoc* examination of an ANN that has already been trained, by inspecting neurons and rationalizing their decisions. A model trained to identify images of dogs might, upon later inspection, be seen to have neurons capturing interpretable properties like 'tail' or 'furry'[38–40]. A limitation of *post hoc* interpretation is that it is disconnected from training, leaving no guarantees as to what level of human understanding can be achieved[41]. Therefore, in attention-based neural networks[42,43], a separate module preselects key 'interpretable' features for input to a black-box model. For example, in a model predicting emotional attitude of a blog author (positive or negative, angry or calm), the key interpretable feature might come from a key phrase preselected from text (LOL, I'm so upset). While DCell has some similarity to these attention-based approaches, its deep hierarchical structure captures many different clusters of features at multiple scales, pushing interpretation from the model input to internal features representing biological subsystems.

In several case studies involving genotypes impacting ER-UPR and DNA-repair subsystems, the subsystem states learned by DCell could be directly confirmed by molecular measurements. Notably, no information about subsystem states was provided during model training. These states emerged from translation of genotypes (model inputs) to growth phenotypes (model outputs) under the structural constraints of the subsystem hierarchy; together, the input–output data and hierarchical structure were sufficient to guide subsystem neurons to learn a biologically correct function. In the future, one might directly supervise a VNN to learn potentially multiple subsystem states and/or complex phenotypes, in which case training data could be provided at any level: genotype, phenotype or points in between.

In some applications of machine learning, predictive performance is all that matters. Indeed, in these cases it is often possible to build a large number of alternative models that, while different in structure, all make excellent near-optimal functional predictions. In biology, however, prediction is not enough. The key additional question is which of the many excellent predictive models is the one actually used by the living system, as optimized not by computation but by evolution. DCell provides proof-of-concept of a system that, while optimizing functional prediction, respects biological structure. Such models are of immediate interest in genome-wide association studies of human disease[44], in which different patient genotypes can influence disease outcomes by complex mechanisms hidden from black-box statistical approaches. Once trained on sufficient data, these models could be applied in personalized therapy by analyzing a patient's genotype in combination with potential points of intervention targeted by drugs. We also see compelling uses in design of synthetic organisms, in which candidate genotypes can be efficiently evaluated *in silico* before validation *in vivo*. Finally, beyond the architecture of the cell, biological systems at other scales may benefit from this type of constrained learning, including modeling of neural connections in the brain.

METHODS

Methods, including statements of data availability and any associated accession codes and references, are available in the online version of the paper.

Note: Any Supplementary Information and Source Data files are available in the online version of the paper.

ACKNOWLEDGMENTS

We gratefully acknowledge support for this work provided by grants from the National Institutes of Health to T.I. (TR002026, GM103504, CA209891, ES014811). We also wish to thank T. Sejnowski and M. Kramer for very helpful comments during development of this work.

AUTHOR CONTRIBUTIONS

J.M., M.K.Y., S.F., R.S. and T.I. designed the study and developed the conceptual ideas. J.M. implemented the main algorithm. M.K.Y. collected all the input sources. J.M. and S.F. implemented all other computational methods and conducted analysis. J.M., M.K.Y., S.F. and T.I. wrote the manuscript with suggestions from the other authors. J.M., M.K.Y., S.F., K.O., E.S. and B.D. designed and developed the server.

COMPETING INTERESTS

T.I. is co-founder of Data4Cure, Inc. and has an equity interest. T.I. has an equity interest in Ideaya BioSciences, Inc. The terms of this arrangement have been reviewed and approved by the University of California, San Diego in accordance with its conflict of interest policies.

1. Farabet, C., Couprie, C., Najman, L. & Lecun, Y. Learning hierarchical features for scene labeling. *IEEE Trans. Pattern Anal. Mach. Intell.* **35**, 1915–1929 (2013).
2. Mikolov, T., Deoras, A., Povey, D., Burget, L. & Černocký, J. Strategies for training large scale neural network language models. In *2011 IEEE Workshop on Automatic Speech Recognition Understanding* 196–201 (IEEE, 2011).
3. Hinton, G. *et al.* Deep neural networks for acoustic modeling in speech recognition: the shared views of four research groups. *IEEE Signal Process. Mag.* **29**, 82–97 (2012).
4. Sainath, T.N., Mohamed, A.R., Kingsbury, B. & Ramabhadran, B. Deep convolutional neural networks for LVCSR. In *2013 IEEE International Conference on Acoustics, Speech and Signal Processing* 8614–8618 (IEEE, 2013).
5. Collobert, R. *et al.* Natural language processing (almost) from scratch. *J. Mach. Learn. Res.* **12**, 2493–2537 (2011).
6. LeCun, Y., Bengio, Y. & Hinton, G. Deep learning. *Nature* **521**, 436–444 (2015).
7. Silver, D. *et al.* Mastering the game of Go with deep neural networks and tree search. *Nature* **529**, 484–489 (2016).
8. Brosin, H.W. An introduction to cybernetics. *Br. J. Psychiatry* **104**, 590–592 (1958).
9. The Gene Ontology Consortium. Expansion of the Gene Ontology knowledgebase and resources. *Nucleic Acids Res.* **45**, D331–D338 (2016).
10. Dutkowski, J. *et al.* A gene ontology inferred from molecular networks. *Nat. Biotechnol.* **31**, 38–45 (2013).
11. Kramer, M., Dutkowski, J., Yu, M., Bafna, V. & Ideker, T. Inferring gene ontologies from pairwise similarity data. *Bioinformatics* **30**, i34–i42 (2014).
12. Carvunis, A.-R. & Ideker, T. Siri of the cell: what biology could learn from the iPhone. *Cell* **157**, 534–538 (2014).
13. Yu, M.K. *et al.* Translation of genotype to phenotype by a hierarchy of cell subsystems. *Cell Syst.* **2**, 77–88 (2016).
14. Copley, S.D. Moonlighting is mainstream: paradigm adjustment required. *BioEssays* **34**, 578–588 (2012).
15. Costanzo, M. *et al.* A global genetic interaction network maps a wiring diagram of cellular function. *Science* **353**, aaf1420 (2016).
16. Costanzo, M. *et al.* The genetic landscape of a cell. *Science* **327**, 425–431 (2010).
17. Szappanos, B. *et al.* An integrated approach to characterize genetic interaction networks in yeast metabolism. *Nat. Genet.* **43**, 656–662 (2011).

18. Lee, I. *et al.* Predicting genetic modifier loci using functional gene networks. *Genome Res.* **20**, 1143–1153 (2010).
19. Pandey, G. *et al.* An integrative multi-network and multi-classifier approach to predict genetic interactions. *PLoS Comput. Biol.* **6**, e1000928 (2010).
20. Xu, C., Wang, S., Thibault, G. & Ng, D.T.W. Futile protein folding cycles in the ER are terminated by the unfolded protein O-mannosylation pathway. *Science* **340**, 978–981 (2013).
21. Free, S.J. Fungal cell wall organization and biosynthesis. *Adv. Genet.* **31**, 33–82 (2013).
22. Walter, P. & Ron, D. The unfolded protein response: from stress pathway to homeostatic regulation. *Science* **334**, 1081–1086 (2011).
23. Scrimale, T., Didone, L., de Mesy Bentley, K.L. & Krysan, D.J. The unfolded protein response is induced by the cell wall integrity mitogen-activated protein kinase signaling cascade and is required for cell wall integrity in *Saccharomyces cerevisiae*. *Mol. Biol. Cell* **20**, 164–175 (2009).
24. Jonikas, M.C. *et al.* Comprehensive characterization of genes required for protein folding in the endoplasmic reticulum. *Science* **323**, 1693–1697 (2009).
25. Srivas, R. *et al.* A UV-induced genetic network links the RSC complex to nucleotide excision repair and shows dose-dependent rewiring. *Cell Rep.* **5**, 1714–1724 (2013).
26. Cadet, J., Sage, E. & Douki, T. Ultraviolet radiation-mediated damage to cellular DNA. *Mutat. Res.* **571**, 3–17 (2005).
27. Pareto, V. *Cours d'Économie Politique* (Librairie Droz, 1964).
28. Farrugia, G. & Balzan, R. Oxidative stress and programmed cell death in yeast. *Front. Oncol.* **2**, 64 (2012).
29. Pujol-Carrion, N. & de la Torre-Ruiz, M.A. Glutaredoxins Grx4 and Grx3 of *Saccharomyces cerevisiae* play a role in actin dynamics through their Trx domains, which contributes to oxidative stress resistance. *Appl. Environ. Microbiol.* **76**, 7826–7835 (2010).
30. Gene Ontology Consortium. Gene Ontology Consortium: going forward. *Nucleic Acids Res.* **43**, D1049–D1056 (2015).
31. Kim, H. *et al.* YeastNet v3: a public database of data-specific and integrated functional gene networks for *Saccharomyces cerevisiae*. *Nucleic Acids Res.* **42**, D731–D736 (2014).
32. Yang, J. *et al.* Genetic variance estimation with imputed variants finds negligible missing heritability for human height and body mass index. *Nat. Genet.* **47**, 1114–1120 (2015).
33. Yang, J., Zaitlen, N.A., Goddard, M.E., Visscher, P.M. & Price, A.L. Advantages and pitfalls in the application of mixed-model association methods. *Nat. Genet.* **46**, 100–106 (2014).
34. Chen, W.W., Niepel, M. & Sorger, P.K. Classic and contemporary approaches to modeling biochemical reactions. *Genes Dev.* **24**, 1861–1875 (2010).
35. Szappanos, B. *et al.* An integrated approach to characterize genetic interaction networks in yeast metabolism. *Nat. Genet.* **43**, 656–662 (2011).
36. Karr, J.R. *et al.* A whole-cell computational model predicts phenotype from genotype. *Cell* **150**, 389–401 (2012).
37. Lipton, Z.C. The mythos of model interpretability. Preprint at https://arxiv.org/abs/1606.03490 (2017).
38. Mahendran, A. & Vedaldi, A. Understanding deep image representations by inverting them. In *Proceedings of the IEEE conference on computer vision and pattern recognition* 5188–5196 (IEEE, 2015).
39. Vondrick, C., Khosla, A., Malisiewicz, T. & Torralba, A. Hoggles: Visualizing object detection features. In *Proceedings of the IEEE International Conference on Computer Vision* 1–8 (IEEE, 2013).
40. Weinzaepfel, P., Jégou, H. & Pérez, P. Reconstructing an image from its local descriptors. In *CVPR 2011* 337–344 (IEEE, 2011).
41. Chakraborty, S. *et al.* Interpretability of deep learning models: a survey of results. Paper presented at IEEE Smart World Congress 2017 Workshop: DAIS 2017, Workshop on Distributed Analytics InfraStructure and Algorithms for Multi-Organization Federations, San Francisco, CA, USA, 7–8 August 2017.
42. Bahdanau, D., Cho, K. & Bengio, Y. Neural machine translation by jointly learning to align and translate. Preprint at https://arxiv.org/abs/1409.0473 (2016).
43. Lei, T., Barzilay, R. & Jaakkola, T. Rationalizing neural predictions. Preprint at https://arxiv.org/abs/1606.04155 (2016).
44. Visscher, P.M., Brown, M.A., McCarthy, M.I. & Yang, J. Five years of GWAS discovery. *Am. J. Hum. Genet.* **90**, 7–24 (2012).

ONLINE METHODS

Preparation of ontologies. We guided the deep neural network structure using a biological ontology consisting of terms representing cellular subsystems, child–parent relations representing containment of one term by another, and gene-to-term annotations. The first ontology considered was the Gene Ontology (GO), in which all three branches of GO (biological process, cellular component, and molecular function) were joined under a single root. We used the following criteria to filter (remove) terms from GO:

1. Terms with the evidence code 'inferred by genetic interaction' (IGI), to avoid potential circularity in predicting genetic interactions in the genotype–phenotype samples.

2. Terms containing fewer than six yeast genes disrupted in the available genotypes (with 'containment' defined as all genes annotated to that term or its descendants).

3. Terms that are redundant with respect to their children terms in the ontology.

When a term was removed, all children were connected directly to all parent terms to maintain the hierarchical structure. The remaining 2,526 terms were used to define the hierarchy of DCell subsystems.

To complement the GO structure, we constructed a data-driven gene ontology using the hierarchical community-detection method of Clique Extracted Ontologies (CliXO) as previously described[11]. Briefly, data on gene pairs were sourced from YeastNet v3 (ref. 31), which lists 68 experimental studies measuring eight different types of molecular interactions (protein–protein interaction, gene coexpression, gene cocitation frequency, and so on), excluding genetic interactions to avoid circularity similar to criterion 1 above. All features were integrated to create a single gene–gene similarity network following a previously described procedure[11], in which each gene–gene pair is assigned a weighted similarity based on a combination of the YeastNet data. This network was subsequently analyzed with the CliXO algorithm, which identifies nested cliques as the threshold gene–gene similarity becomes progressively less stringent. This process yields a hierarchy (directed acyclic graph) of parent–child relations among 4,805 cliques at different similarity thresholds, with each clique representing a cellular subsystem. By performing an ontology alignment[10] between the CliXO and GO hierarchies, we found that 1,811 (38%) of the CliXO subsystems contained a significantly overlapping (FDR < 0.1) set of genes with GO terms. We further filtered the CliXO hierarchy using the same procedure for filtering GO (removing subsystems containing fewer than six yeast genes disrupted in the available genotypes). The remaining 713 subsystems were used to define the DCell hierarchy.

DCell architecture and training algorithm. DCell trains a deep neural network to predict phenotype from genotype, with architecture that exactly mirrors the hierarchical structure of an ontology of cellular subsystems. Each cellular subsystem is represented by a group of hidden variables (neurons) in the neural network, and each parent–child relation is represented by a set of edges that fully connect these groups of hidden variables. The depth of this architecture (12 layers) presents two challenges for training: (i) There is no guarantee that each subsystem will learn new patterns instead of copying those of its child subsystems; and (ii) gradients tend to vanish lower in the hierarchy. To tackle these challenges, we borrow ideas from two previous systems, GoogLeNet[45] and Deeply-Supervised Net[46], which improve the transparency and discriminative power of hidden variables and reduce the effect of vanishing gradients.

We denote our input training data set as $D = \{(X_1, y_1), (X_2, y_2),\ldots, (X_N, y_N)\}$, where N is the number of samples. For each sample i, $X_i \in R^M$ denotes the genotype, represented as a binary vector of states on M genes (1 = disrupted; 0 = wild type), and $y_i \in R$ denotes the observed phenotype, which can be either relative growth rate or genetic interaction value. The multidimensional state of each subsystem t, denoted by the output vector $O_i^{(t)}$, is defined by a nonlinear function of the states of all of its child subsystems and annotated genes, concatenated in the input vector $I_i^{(t)}$:

$$O_i^{(t)} = BatchNorm(Tanh(Linear(I_i^{(t)}))) \quad (1)$$

$Linear(I_i^{(t)})$ is a linear transformation of $I_i^{(t)}$ defined as $W^{(t)}I_i^{(t)} + b^{(t)}$. Let $L_O^{(t)}$ denote the length of $O_i^{(t)}$, representing the number of values in the state of t and determined by

$$L_o^{(t)} = \max\left(20, \left\lceil 0.3 * \text{number of genes contained by } t \right\rceil\right) \quad (2)$$

Intuitively, larger subsystems have larger state vectors to capture potentially more complex biological responses. Similarly, let $L_I^{(t)}$ denote the length of $I_i^{(t)}$. In equation (1), $W^{(t)}$ is a weight matrix with dimensions $L_O^{(t)} \times L_I^{(t)}$, and $b^{(t)}$ is a column vector with size $L_O^{(t)}$. $W^{(t)}$ and $b^{(t)}$ provide the parameters to be learned for subsystem t. *Tanh* is the nonlinear transforming hyperbolic tangent function. *Batch Norm*[47] is a normalizing function that reduces the impact of internal covariate shift caused by different scales of weights in $W^{(t)}$. Batch normalization can be viewed as a type of regularization of model weights and reduces the need for the traditional dropout step in deep learning. We perform the training process by minimizing the objective function:

$$\frac{1}{N}\sum_{i=1}^{N}(Loss(Linear(O_i^{(r)}), y_i) + \alpha\sum_{t\neq r} Loss(Linear(O_i^{(t)}), y_i)) + \lambda \| W \|_2 \quad (3)$$

Here, *Loss* is the squared error loss function, and r is the root of the hierarchy. Note that we compare y_i with not only the root's output, $O_i^{(r)}$, but also the outputs of all other subsystems, $O_i^{(t)}$. *Linear* in equation (3) denotes linear functions transforming multidimensional vector $O_i^{(t)}$ into a scalar. In this way, every subsystem is optimized to serve its parents as features and to predict the phenotype itself, as used previously by GoogLeNet[45]; the parameter α (=0.3) balances these two contributions. λ is an l_2 norm regularization factor determined by four-fold cross-validation. To train the DCell model, we initialize all weights uniformly at random between −0.001 and 0.001. We optimize the objective function using ADAM[48], a popular stochastic gradient descent algorithm, with mini-batch size of 15,000. Gradients with respect to model parameters are computed by standard back propagation[49]. Note that while other hyperparameters might influence the overall predictive performance, they are unrelated to our focus on biological interpretation as long as the same settings are applied to both DCell and the black-box models we use as controls (**Fig. 2d**). We implemented DCell using the Torch7 library (https://github.com/torch/torch7) on Tesla K20 GPUs.

Training genotype–phenotype data. Several forms of the model were employed in this study, trained on either Costanzo *et al.*[16] (~3 million training examples) or a more recently published update in 2016 (~8 million training examples)[15]. The first model was used for all results and figures in the main text to enable comparisons against previous approaches to predict genetic interactions. The latter model with updated data is provided at d-cell.ucsd.edu.

Alternative genotype–phenotype translation methods. We compared DCell to three state-of-the-art nonhierarchical approaches for predicting genetic interactions: flux balance analysis (FBA)[35], multinetwork multiclassifier (MNMC)[19], and guilt-by-association (GBA)[18]. FBA uses a model of metabolism to assess the impact on cell growth of gene deletions in metabolic pathways. MNMC is an ensemble supervised learning system that uses many different data sets as features to predict genetic interactions. GBA predicts the genetic interaction score of pairwise gene deletions based on the phenotypes of their network neighbors. We also compared against our previous prediction method (Ontotype)[13] which applies prior knowledge from a hierarchy like GO or CliXO but does not use deep learning nor simulate the internal states of subsystems. Ontotype counts the number of genes knocked in every GO term and uses these counts as features in a random forest regression.

Relative local improvement in predictive power. The RLIPP score was used to quantify and compare the importance of DCell's internal subsystems in prediction of phenotype. To calculate the RLIPP score of a subsystem, we compared two different linear models for phenotypic prediction. In the first model, the subsystem's neurons were used as features in an l_2-norm penalized linear regression (**Supplementary Fig. 3a**). In the second model, the neurons of the subsystem's children were used as the features instead. Each model was trained separately, with the optimal hyperparameter associated with the l_2-norm penalty determined in five-fold cross-validation. The performance of each of these two models was calculated as the Spearman correlation between the predicted and measured phenotype, here taken as genetic interaction scores (**Supplementary Fig. 3b,c**). The RLIPP score was defined as the performance of the parent model relative to that of the children (**Supplementary Fig. 3d**). A positive RLIPP score indicates that the state of the parent subsystem is more predictive of phenotype than the states of its children. This situation can occur when the parent learns complex (nonlinear) patterns from the children, as opposed to merely copying or adding their values. The intuition behind the RLIPP score is similar to a related 'linear probe' technique developed in a previous study to characterize the utility of each layer of a deep neural network[50].

Identification of subsystems that mimic Boolean logic gates. As one means to interpret the mechanisms by which DCell translates genotype to phenotype, we evaluated each subsystem for the extent to which it approximates Boolean logic. In particular, we considered all trios of subsystems, each consisting of a parent subsystem and two of its children, and tested whether their binary states (S,C_1,C_2) were well approximated by nontrivial Boolean logic (**Supplementary Table 1**). For each genotype, the binary state of each child subsystem was defined as either 'Wild Type' (True) or 'Disrupted' (False) by comparing PC1 to the wild-type state. The binary state of each parent subsystem was defined as either '≤Wild Type' (True) or '>Wild Type' (False, by comparing PC1 to the wild-type state. For each combinatorial state (C_1,C_2) of two child subsystems, the parent state S implied by DCell was determined based on the majority parent states of genotypes annotated to (C_1,C_2). For instance, suppose that for all the genotypes that induce (C_1 = True, C_2 = False) in the two children, DCell transforms 80% to parent state S = True and 20% to state S = False. We conclude the underlying logic for the parent subsystem to translate the signal from children subsystems is (True, False)→True. By checking the parent states for all four possible (C_1,C_2) combinations, we can decide whether this trio of subsystems exhibits Boolean logic (**Supplementary Table 1**). A trio belongs to none of the logic functions if >50% of all the genotypes or <4 genotypes are annotated to any (C_1,C_2) combinatorial state, or none of the annotated genotypes yield significant genetic interactions ($|\varepsilon| <= 0.08$). For those subsystems exhibiting Boolean logic, we excluded 'trivial' functions in which the parent is always True, always False, or follows one of the children without dependence on the other.

DCell server construction. The DCell server (http://d-cell.ucsd.edu/) comprises several interconnected components working in unison to collect user input, run simulations, and transcode results to the web interface. On the backend, the DCell neural network model runs on the Torch library on a dedicated multi-GPU machine. On the front end, the web interface is built on cytoscape.js[51] and an in-house D3 (ref. 52) graph visualizer to display a subgraph of the hierarchy, and React[53] for agile DOM (Document Object Model) editing[54]. To respond to user input, including searching and viewing details of model subsystems, a low-latency proxy service translates between plain text fetched from the front end and binary data used by the backend. An Elasticsearch cluster[55] caches and indexes data for fast lookup and predictions. All web services run on a Kubernetes-based cloud infrastructure (http://kubernetes.io/) that autoscales to heavy workloads. The result of these efforts is to allow easy visualization and interactivity of the model.

Life Sciences Reporting Summary. Further information on experimental design is available in the **Life Sciences Reporting Summary**.

Data availability. The D-Cell server is available at http://d-cell.ucsd.edu/. The software implementation and dataset are available at https://github.com/idekerlab/DCell/.

45. Szegedy, C. *et al.* Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition* 1–9 (IEEE, 2015).
46. Lee, C.-Y., Xie, S., Gallagher, P.W., Zhang, Z. & Tu, Z. *Deeply-Supervised Nets. in AISTATS* **2**, 5 (2015).
47. Ioffe, S. & Szegedy, C. Batch normalization: accelerating deep network training by reducing internal covariate shift. Preprint at https://arxiv.org/abs/1502.03167 (2015).
48. Kingma, D.P. & Ba, J. Adam: a method for stochastic optimization. Preprint at https://arxiv.org/abs/1412.6980 (2017).
49. Rumelhart, D.E., Hinton, G.E. & Williams, R.J. Learning representations by back-propagating errors. *Nature* **323**, 533–536 (1986).

doi:10.1038/nmeth.4627

50. Alain, G. & Bengio, Y. Understanding intermediate layers using linear classifier probes. Preprint at https://arxiv.org/abs/1610.01644 (2016).
51. Franz, M. *et al.* Cytoscape.js: a graph theory library for visualisation and analysis. *Bioinformatics* **32**, 309–311 (2016).
52. Bostock, M., Ogievetsky, V. & Heer, J. D^3: data-driven documents. *IEEE Trans. Vis. Comput. Graph.* **17**, 2301–2309 (2011).
53. Stefanov, S. *React: Up & Running: Building Web Applications.* (O'Reilly Media, 2016).
54. Wood, L., Nicol, G., Robie, J., Champion, M. & Byrne, S. Document Object Model (DOM) level 3 core specification. *W3C* https://www.w3.org/TR/2004/REC-DOM-Level-3-Core-20040407/DOM3-Core.html. (2004).
55. Gormley, C. & Tong, Z. *Elasticsearch: The Definitive Guide: A Distributed Real-Time Search and Analytics Engine* (O'Reilly Media, 2015).

nature research

Corresponding author(s): Trey Ideker

Life Sciences Reporting Summary

Nature Research wishes to improve the reproducibility of the work that we publish. This form is intended for publication with all accepted life science papers and provides structure for consistency and transparency in reporting. Every life science submission will use this form; some list items might not apply to an individual manuscript, but all fields must be completed for clarity.

For further information on the points included in this form, see Reporting Life Sciences Research. For further information on Nature Research policies, including our data availability policy, see Authors & Referees and the Editorial Policy Checklist.

Please do not complete any field with "not applicable" or n/a. Refer to the help text for what text to use if an item is not relevant to your study.
For final submission: please carefully check your responses for accuracy; you will not be able to make changes later.

Experimental design

1. Sample size

   Describe how sample size was determined. N/A

2. Data exclusions

   Describe any data exclusions. N/A

3. Replication

   Describe the measures taken to verify the reproducibility of the experimental findings. N/A

4. Randomization

   Describe how samples/organisms/participants were allocated into experimental groups. N/A

5. Blinding

   Describe whether the investigators were blinded to group allocation during data collection and/or analysis. N/A

   Note: all in vivo studies must report how sample size was determined and whether blinding and randomization were used.

6. Statistical parameters

   For all figures and tables that use statistical methods, confirm that the following items are present in relevant figure legends (or in the Methods section if additional space is needed).

n/a	Confirmed	
☒	☐	The exact sample size (*n*) for each experimental group/condition, given as a discrete number and unit of measurement (animals, litters, cultures, etc.)
☒	☐	A description of how samples were collected, noting whether measurements were taken from distinct samples or whether the same sample was measured repeatedly
☒	☐	A statement indicating how many times each experiment was replicated
☐	☒	The statistical test(s) used and whether they are one- or two-sided *Only common tests should be described solely by name; describe more complex techniques in the Methods section.*
☐	☒	A description of any assumptions or corrections, such as an adjustment for multiple comparisons
☐	☒	Test values indicating whether an effect is present *Provide confidence intervals or give results of significance tests (e.g. P values) as exact values whenever appropriate and with effect sizes noted.*
☐	☒	A clear description of statistics including central tendency (e.g. median, mean) and variation (e.g. standard deviation, interquartile range)
☐	☒	Clearly defined error bars in all relevant figure captions (with explicit mention of central tendency and variation)

See the web collection on statistics for biologists for further resources and guidance.

Software

Policy information about availability of computer code

7. Software

Describe the software used to analyze the data in this study.	We provide the code for our method and clearly direct readers to the Git repository to access our code.

For manuscripts utilizing custom algorithms or software that are central to the paper but not yet described in the published literature, software must be made available to editors and reviewers upon request. We strongly encourage code deposition in a community repository (e.g. GitHub). *Nature Methods* guidance for providing algorithms and software for publication provides further information on this topic.

Materials and reagents

Policy information about availability of materials

8. Materials availability

Indicate whether there are restrictions on availability of unique materials or if these materials are only available for distribution by a third party.	N/A

9. Antibodies

Describe the antibodies used and how they were validated for use in the system under study (i.e. assay and species).	N/A

10. Eukaryotic cell lines

a. State the source of each eukaryotic cell line used.	N/A
b. Describe the method of cell line authentication used.	N/A
c. Report whether the cell lines were tested for mycoplasma contamination.	N/A
d. If any of the cell lines used are listed in the database of commonly misidentified cell lines maintained by ICLAC, provide a scientific rationale for their use.	N/A

Animals and human research participants

Policy information about studies involving animals; when reporting animal research, follow the ARRIVE guidelines

11. Description of research animals

Provide all relevant details on animals and/or animal-derived materials used in the study.	N/A

Policy information about studies involving human research participants

12. Description of human research participants

Describe the covariate-relevant population characteristics of the human research participants.	N/A