SI Appendix Materials and Methods

Preparation of the Gold Standard Dataset for DeepEC. A gold standard dataset was prepared by processing 253,255 protein sequences assigned with EC numbers from the Swiss-Prot (1) dataset (covering a total of 559,542 protein sequences; released February 2018) and 11,462,777 protein sequences with EC numbers from the TrEMBL (1) dataset (covering a total of 11,462,777 protein sequences; released February 2018) through a few steps as described below (*SI Appendix,* Fig. S1*A*-*C*). Both Swiss-Prot and TrEMBL datasets were obtained from UniProt (2). Only the protein sequences having EC numbers were retrieved from both datasets.

Only the Swiss-Prot dataset was initially considered as a gold standard dataset because it has manually curated EC numbers for the protein sequences. However, among a total of 4,669 EC numbers covered by the Swiss-Prot dataset, 2,945 EC numbers were initially covered by fewer than 10 protein sequences each, which are not sufficient to properly train a machine learning model for a specific EC number (*SI Appendix,* Fig. S2). In order to overcome this problem, protein sequences from the TrEMBL dataset were additionally considered for the gold standard dataset. However, because the TrEMBL dataset has protein sequences with automatically generated EC numbers that can be erroneous (3), only the TrEMBL protein sequences with high confidence annotations were selected for the gold standard dataset through sequence alignment with optimally determined parameters (*SI Appendix,* Fig. S1). This process increased the number of EC numbers that are covered by at least 10 protein sequences among the 4,669 EC numbers in the resulting gold standard dataset (*SI Appendix,* Fig. S2). The final version of the gold standard dataset used in this study covers 1,388,606 protein sequences and 4,669 EC numbers where 2,429 EC number are covered by at least 10 protein sequences, and 2,240 EC numbers are covered by fewer than 10 protein sequences.

Finally, because a CNN in general accepts input data with a fixed size (i.e., protein sequence with a fixed number of amino acids in this study), protein sequences having 1,000

www.pnas.org/cgi/doi/10.1073/pnas.1821905116

amino acids or fewer in the gold standard dataset were only considered for the development of DeepEC (4). As a result, size of the gold standard dataset was reduced from 1,388,606 to 1,360,727 protein sequences to meet this upper limit. The number of EC numbers in the reduced gold standard dataset remain the same as 2,429. This upper limit of 1,000 amino acids for a protein sequence was considered to be an optimal trade-off point between the number of protein sequences used for a CNN training and computational complexity (4). Use of a smaller number of amino acids as the upper limit reduces the number of protein sequences in the gold standard dataset that can be used to train CNNs. Meanwhile, use of a greater number of amino acids as the upper limit significantly increases the computational complexity. It should be noted that protein sequences having more than 1,000 amino acids correspond to 2.0% (i.e., 27,837 protein sequences) of the gold standard dataset covering 1,360,727 protein sequences. DeepEC can also predict EC numbers for a protein sequence with more than 1,000 amino acids by iterating the prediction of EC numbers for a series of regions (each having 1,000 amino acids) of a given protein sequence, and by combing the classification outcomes (i.e., predicted EC numbers). For the 27,837 protein sequences belonging to 404 EC numbers, which have more than 1,000 amino acids, and thus have been removed from the gold standard dataset, 9,843 protein sequences were predicted to have at least one EC number. Among the 9,843 protein sequences, 7,892 protein sequences were correctly assigned with EC numbers by DeepEC.

For the development of DeepEC, the resulting gold standard dataset (1,360,727 protein sequences) was split into three datasets, namely training (70%), validation (10%), and testing (20%). CNN modeling (i.e., training, validation and testing) procedure is described below.

Optimizing the Architecture of DeepEC. DeepEC consists of three independent CNNs in order to perform three different classification tasks for a single protein sequence given as an input. The three CNNs share the same ‘embedding’, ‘convolution’, and ‘1-max pooling layers’,

but have different 'fully connected layers' (Fig. 1). Structure of each CNN was determined by examining four main hyperparameters (4, 5), including the window size of each filter, number of filters for each window size, number of hidden layers and number of nodes in each hidden layer of the fully connected layer (Dataset S1). Values for the four main hyperparameters that generated the highest macro precision were used to model the CNN-1, CNN-2 and CNN-3. All the CNN-1, CNN-2 and CNN-3 performed the best using the following hyperparameter values: three different types of filters (i.e., sizes of 4 × 21, 8 × 21 and 16 × 21) with a stride of 'one', 128 filters for each size, and two hidden layers with 512 hidden nodes in the fully connected layer (Fig. 1 and Dataset S1). For other parameters, default values set in a Python package *Keras* (version 2.1.6) were used.

In an embedding layer, an embedding matrix having a dimension of 1,000 (a maximum length of a protein sequence to detect, i.e., 1,000 amino acids) × 21 (20 standard amino acids and an unknown amino acid 'X') is generated from an input protein sequence using one-hot encoding method (4, 6) (Fig. 1). While the embedding matrix has a fixed size (1,000 × 21), an input protein sequence can have any size within the scope of the embedding matrix (i.e., 1,000 amino acids).

In a convolutional layer, a 'filter' is a two-dimensional array of random weights ('W'), and three different filters, each having different sizes (4 × 21, 8 × 21 and 16 × 21), were considered in this study. For each size of the filter, 128 different filters were considered in this study, each having a different set of weights. With this in mind, convolution of the embedding matrix with a filter (having one of the three sizes and a set of random weights) is implemented to generate a 'convolved value', which is equivalent to $W \cdot x + b$. This convolved value serves as an input for an activation function, a rectified linear unit (ReLU) in this study. Output of the activation function ReLU is an 'activation value' (small red boxes in 'Convolution and 1-max pooling layers' in Figure 1). As the filter continues to convolve the entire embedding matrix

with a stride of one, corresponding activation values are generated, and make up an 'activation map' as a result (Fig. 1). This procedure of creating an activation map is repeated with filters with the same size, but each having a different set of random weights (i.e., a total of 128 sets in this study as mentioned above). In this study, 128 activation maps are created for each size of the filter, and hence 128 different filters for a single size. This process is repeated using filters with three different sizes (4 × 21, 8 × 21 and 16 × 21) in this study. These activation maps contain information on important features (e.g., domains) of an input protein sequence (7).

In an 1-max pooling layer, one maximum value is selected from each activation map, which subsequently forms an 'univariate feature vector' (Fig. 1). Finally, a fully connected layer receives a processed univariate feature vector to conduct one of the classification tasks mentioned above.

Finally, as described in the main text, homology analysis is additionally implemented in DeepEC if an input protein sequence fails to be assigned with EC number(s) using the abovementioned three CNNs. Homology analysis is implemented using DIAMOND (8) (Fig. 1). The input protein sequence becomes a query sequence in this homology analysis, and is assigned with EC number(s) from a homologous protein annotated with EC number(s) in the gold standard dataset (*SI Appendix,* Fig. S1*A-C*).

Training CNNs of DeepEC. Weights in DeepEC were randomly initialized using a uniform distribution. ReLU was used as an activation function for the convolution layer and hidden layers of the fully connected layer (i.e., a final stage of each CNN within DeepEC).

$$f(x) = \max(0, x)$$

Sigmoid function was used as an activation function for an output layer of the fully connected layer.

$$f(x) = \frac{1}{1 + e^{-x}}$$

Batch normalization was used to reduce the internal-covariate-shift by normalizing input distribution of each layer to a standard Gaussian distribution (9). It should be noted that batch normalization also functions as a regularizer, which helps avoid overfitting of a deep learning model. DeepEC was trained by minimizing prediction errors in comparison with the gold standard dataset using cross entropy as a loss function and Adam method for optimization (10). CNN-1, CNN-2 and CNN-3 of the DeepEC were trained up to 30 epochs, and those showing the lowest test loss value across the 30 epochs were selected for the use in DeepEC to ensure that these three CNNs were not overfitted (*SI Appendix,* Fig. S3).

Validating and Testing CNNs of DeepEC. After the DeepEC training, the performance was validated and tested using the validation and testing datasets (see the section above) and the following two performance metrics:

$$Macro\ precision\ = \frac{\sum_{i=1}^{l} \frac{TPi}{TPi + FPi}}{l}$$

$$Macro\ recall = \frac{\sum_{i=1}^{l} \frac{TPi}{TPi + FNi}}{l}$$

where *TP*, *TN*, *FP* and *FN* indicate the number of true positives, true negatives, false positives and false negatives, respectively, and *l* is the number of labels (i.e., number of EC numbers).

Prediction of EC Numbers for All the Enzymes from 9,513 Complete Genomes Using DeepEC. To predict EC numbers using DeepEC, protein sequences from 9,513 complete genomes available at NCBI Genome database (https://www.ncbi.nlm.nih.gov/genome) were downloaded from NCBI FTP (ftp://ftp.ncbi.nih.gov/). Protein sequences in a FASTA format for

each genome were used as inputs for DeepEC.

Preparation of Enzyme and Nonenzyme Protein Sequences for the Modeling of CNN-1. For the modeling of CNN-1 that classifies whether a protein sequence is an enzyme or a non-enzyme, 153,884 enzyme and non-enzyme protein sequences were prepared using the gold standard dataset. First, 153,884 enzyme protein sequences belonging to the Swiss-Prot dataset were retrieved from the gold standard dataset. Next, the same number of non-enzyme protein sequences were prepared by oversampling 22,168 non-enzyme protein sequences obtained from DEEPre (11).

Preparation of Protein Sequences for the L-Alanine Scanning Method that Mutates Domains and Binding Site Residues. For a systematic comparison among the six different EC number prediction tools (i.e., DeepEC, CatFam, DETECT v2, ECPred, EFICAz2.5 and PRIAM), the 2,435 input protein sequences were prepared as follows. First, 487 EC numbers that can be commonly classified by these tools were defined. Protein sequences in the gold standard dataset were considered as inputs for DeepEC only if they are assigned with one of these 487 EC numbers. Next, five protein sequences were randomly retrieved from the gold standard dataset for each of the 487 EC numbers. This is to ensure an equal coverage of all the EC numbers with protein sequences. As a result, 2,435 protein sequences were prepared as inputs for the implementation of these six different EC number prediction tools, which correspond to five protein sequences for each of the 487 EC numbers. Next, domains of the 2,435 protein sequences were detected by using PfamScan (12), and subjected to the L-alanine scanning method (13).

For mutating binding site residues of protein sequences, structural information on 1,134 enzymes bound with natural substrates was first obtained from RCSB PDB database (14).

Residues were determined to be binding site residues if they were within 4 Å distance from the location of each substrate bound with an enzyme (15). All the binding site residues were subsequently substituted with L-alanine residues.

Expression and Purification of a Putative L-Threonate Dehydrogenase. A plasmid pET30a-his-ygbJ expressing the his-tagged *ygbJ* was constructed by amplifying the wild-type *ygbJ* gene from genomic DNA of *Escherichia coli* W3110 using the following forward and reverse primers, and by inserting the *ygbJ* gene to a plasmid pET30a (Km^R, an empty expression plasmid, T7 promoter, pBR322 ori, 5.4 kb; Merck Millipore, Burlington, MA) through *Nde*I and *EcoR*I sites.

- Forward primer:

  5'–AGACAGGAATTCTAAGAAGGAGATATAATGCATCATCACCATCACCAC

  AAAACGGGATCTGAGTTTCATGTC–3'
- Reverse primer:

  5'–AGACAGGTCGACTCATGATTTCGCTCCCGGT –3'

For expression of the N-terminus his-tagged *ygbJ*, *E. coli* BL21 (DE3) harboring pET30a-his-ygbJ was cultured in 200 mL of LB medium at 37 °C. When the optical density at 600 nm (OD_{600}) reached 0.6, IPTG was added at a final concentration of 0.2 mM for induction of the his-tagged *ygbJ* gene. After additional 4 h of cultivation, cells were lysed with a lysis buffer containing 100 mM sodium carbonate (pH 10.0). Soluble fractions of the cell lysates were dialyzed using buffer A containing 20 mM Tris-HCl and 150 mM NaCl (pH 8.0). To purify the N-terminal His-tagged YgbJ, chromatography was performed using a XK 16 mm column (GE Healthcare, Chicago, IL) with Ni-NTA resin (8 mL). Proteins bound to the resin

were gradually eluted using 25% buffer B containing 20 mM Tris-HCl, 150 mM NaCI, and 1M imidazole (pH 8.0). The eluted sample was subsequently washed using the buffer A. The purified YgbJ protein was quantified by using the Bio-Rad Protein Assay Kit (Bio-Rad, Hercules, CA) with bovine serum albumin (BSA) as a standard.

***In Vitro* YgbJ Assay.** Enzyme reaction was carried out by using 150 μg of the purified His-tagged YgbJ with 0.1 mM NAD and 5 mM of a substrate (D-glycerate or L-threonate) in phosphate-buffered saline (1 mL for a total reaction volume) for 10 min at room temperature. Enzyme activity was determined by measuring the total NADH level (pM) using a NAD/NADH quantitation colorimetric kit (Cat# K337-100; BioVision, Milpitas, CA). Each well of the 96-well plate was read using a spectrophotometer at OD_{450}. For absolute quantification of the NADH concentration, a standard curve was prepared according to the manufacturer's protocol.

Development Environment. Modeling CNNs was implemented using a Python package *Keras* (version 2.1.6) (https://keras.io/) with *TensorFlow* backend (version 1.5.0) (16). Python package *scikit-learn* (17) was used to calculate precision and recall values for the EC numbers predicted using DeepEC. A workstation with dual Intel Xeon E5-2670 v3 (24 core processors, 2.3 GHz each) and NVIDIA GeForce GTX 1080 graphics-processing unit was used for the model training.

SI Appendix Figures

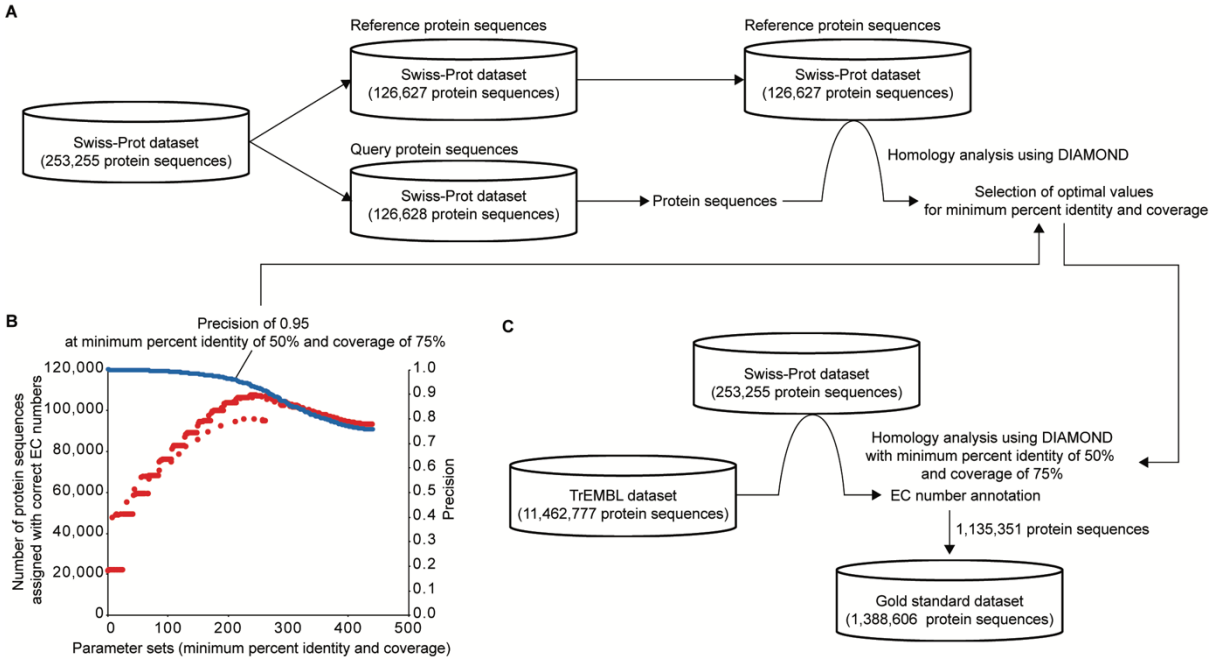

Fig. S1. Preparation of the gold standard dataset using Swiss-Prot and TrEMBL datasets for the development of DeepEC. (*A*) Optimal parameters for the sequence alignment were first examined using the Swiss-Prot dataset. For this, 253,255 protein sequences in the Swiss-Prot (1) dataset were split into two groups with an approximately equal size. Protein sequences from one group were used as queries, and those from the other group were used as reference protein sequences. (*B*) Query protein sequences were aligned with reference protein sequences by using varied values (5% - 95%) for each of the two parameters, minimum percent identity and coverage (parameter sets on the *x*-axis), while fixing all the other parameters at default values. Sequence alignment was conducted by using DIAMOND (version 0.9.22), which is a sequence aligner for DNA and protein sequences (8). Query protein sequence was assigned with EC numbers from its homologous protein sequence, based on the sequence alignment, which already has EC numbers in a database. When using a minimum percent identity of 50% and coverage of 75%, 95% EC numbers were correctly predicted (i.e., precision of 0.95). Precision of 0.95 was considered to be ideal for further homology analysis via sequence alignment. Use

of a greater cutoff value of precision (blue dots) significantly reduced the number of protein sequences (red dots) in the TrEMBL dataset that are to be added to the gold standard dataset (see *SI Appendix, Materials and Methods*). (*C*) Using the parameters obtained from (*A*) and (*B*), protein sequences from the TrEMBL (1) dataset were aligned with those from the Swiss-Prot dataset. The TrEMBL protein sequences found to have homologous protein sequences in the Swiss-Prot dataset were newly added to the gold standard dataset. By so doing, 2,429 EC numbers in the gold standard dataset are covered by at least 10 protein sequences each. Ten protein sequences are the minimally required number of data samples to properly train a machine learning model for a specific EC number (*SI Appendix,* Fig. S2).

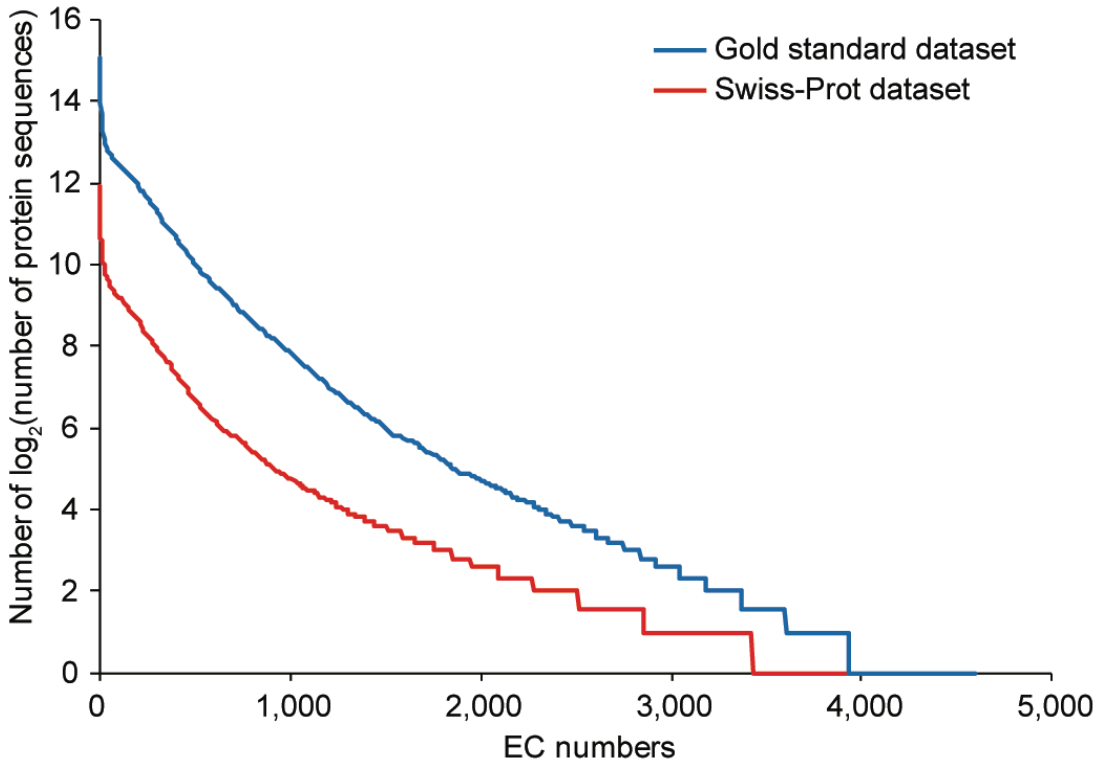

Fig. S2. Number of protein sequences associated with each EC number (*x*-axis) in the Swiss-Prot dataset and gold standard dataset.

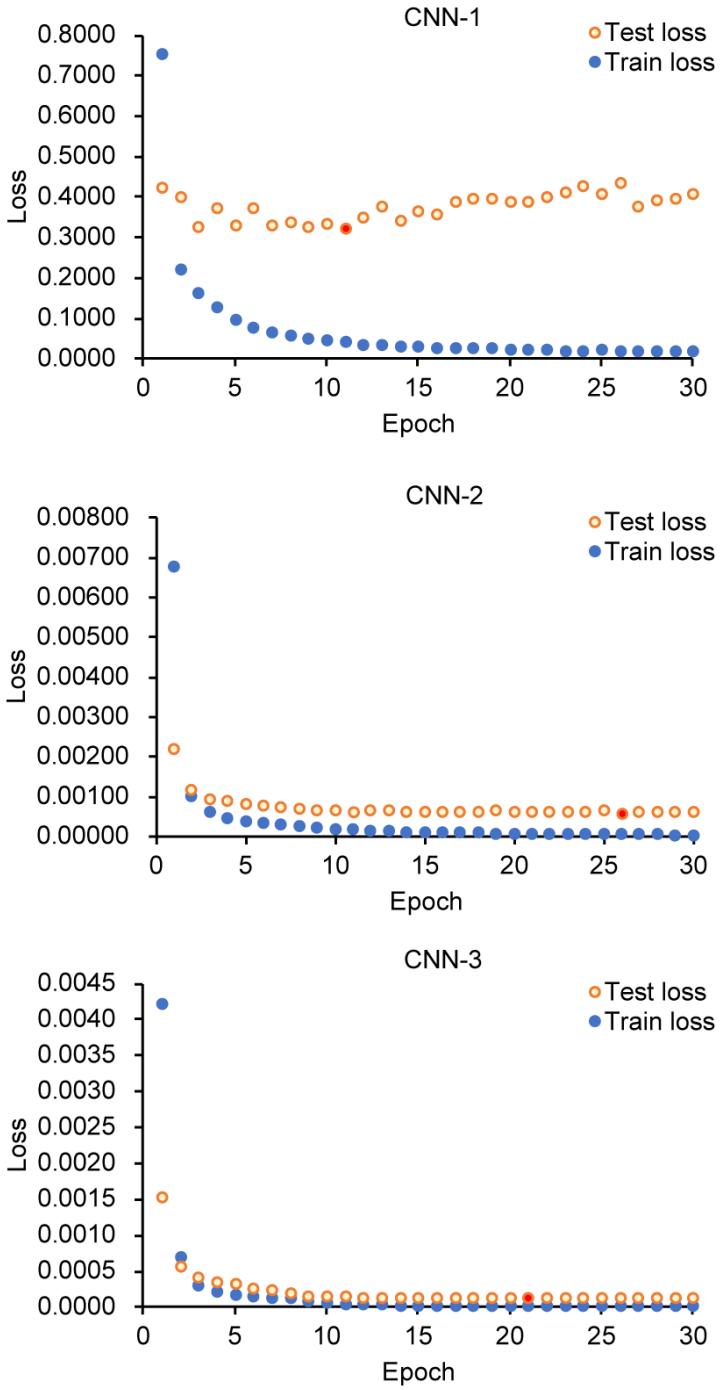

Fig. S3. Test and train loss curves of CNN-1, CNN-2, and CNN-3, with all their four main hyperparameters (i.e., the window size of each filter, number of filters for each window size, number of hidden layers and number of nodes in each hidden layer of the fully connected layer) optimized, for 30 epochs. Test and train loss values of each CNN were obtained from each epoch (*SI Appendix, Materials and Methods,* and Dataset S1). CNN-1, CNN-2, and CNN-3

showing the lowest test loss value (red dot) across the 30 epochs were selected for the use in DeepEC.

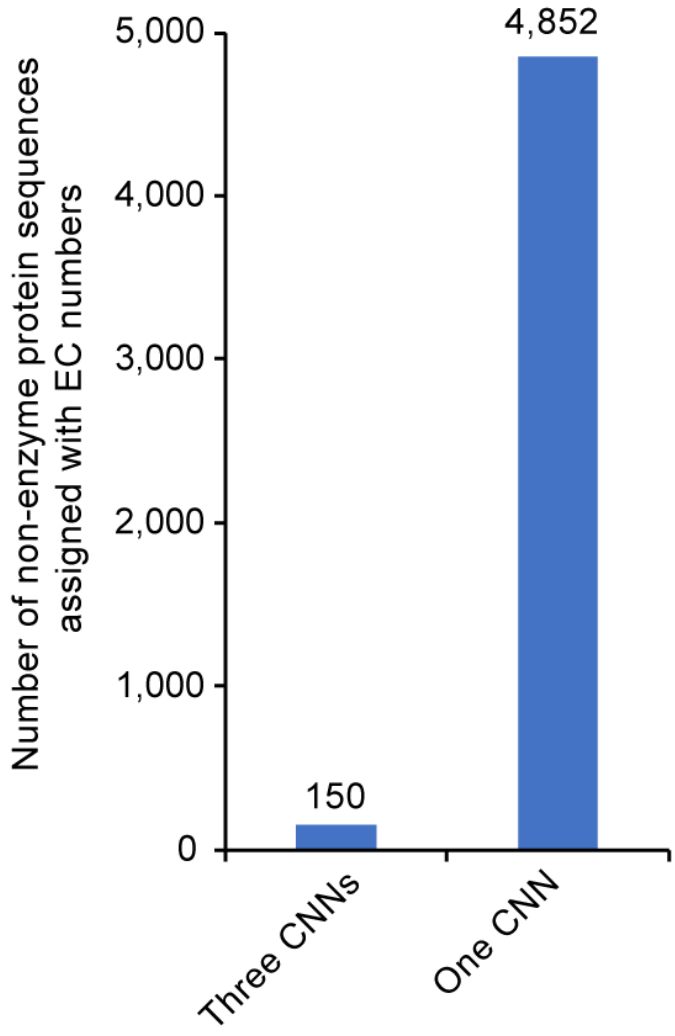

Fig. S4. Number of 22,168 non-enzyme protein sequences assigned with EC numbers when using three CNNs and one CNN in DeepEC. EC numbers of 22,168 non-enzyme protein sequences obtained from DEEPre (11) were predicted using two different versions of DeepEC, having three CNNs and one CNN. Each of the three CNNs in DeepEC conducts a single classification task: binary classification of a given protein sequence as an enzyme or a non-enzyme protein, and prediction of the third-level and fourth-level EC numbers. A single CNN was designed to conduct the abovementioned multiple classification tasks by itself.

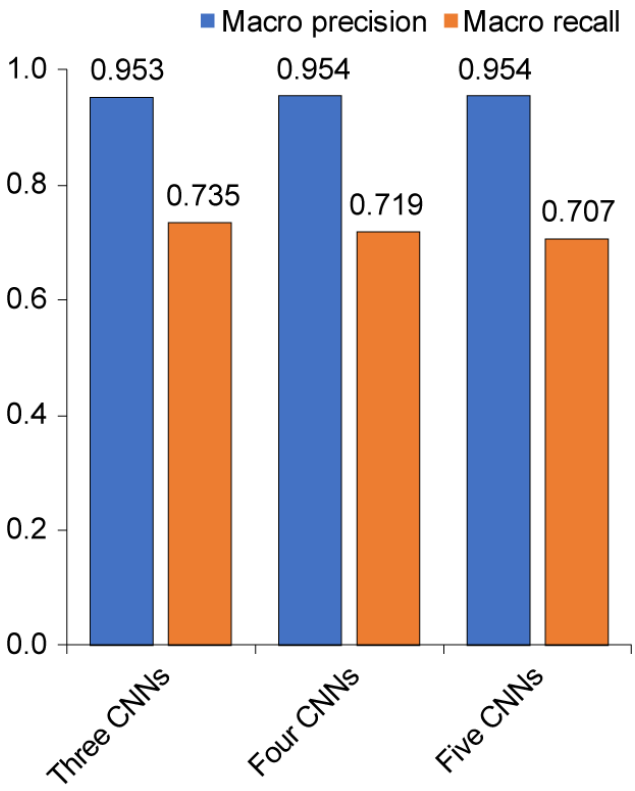

Fig. S5. Prediction performance of three different versions of DeepEC having three, four and five CNNs. 'Three CNNs' herein refers to DeepEC having three independent CNNs that classify whether an input protein sequence is enzyme or not (using 'CNN-1'), and predict the third-level and fourth-level EC numbers (using 'CNN-2' and 'CNN-3', respectively). 'Four CNNs' refers to the DeepEC having an additional CNN that predicts second-level EC numbers. 'Five CNNs' refers to the DeepEC having two additional CNNs, each predicting the first-level and second-level EC number, respectively. For each version of DeepEC (i.e., use of three, four and five CNNs), EC number(s) are generated as output for a given protein sequence only if all the CNNs generate consistent results. As a result of the analysis, three CNNs were used in DeepEC because this version achieved the greatest recall value with its precision value barely compromised, compared with the other two versions having four and five CNNs. Protein sequences in the 'testing dataset' of the gold standard dataset (see *SI Appendix, Materials and Methods*) were used as inputs for this analysis.

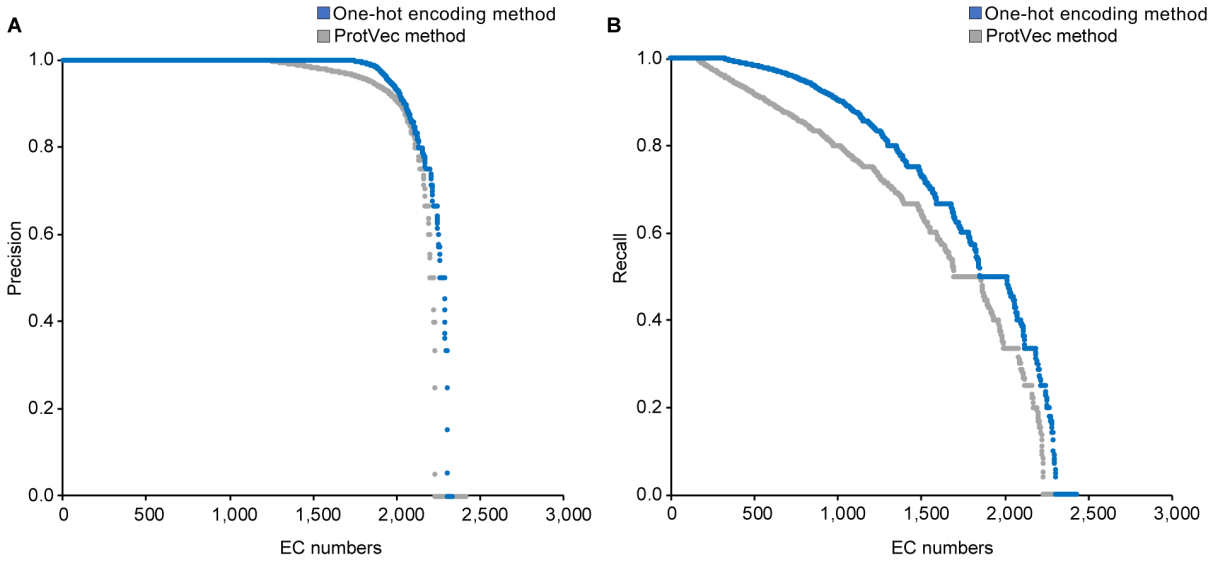

Fig. S6. Prediction (*A*) and recall (*B*) values for the EC numbers predicted using two different versions of DeepEC using one-hot encoding method or ProtVec (11).

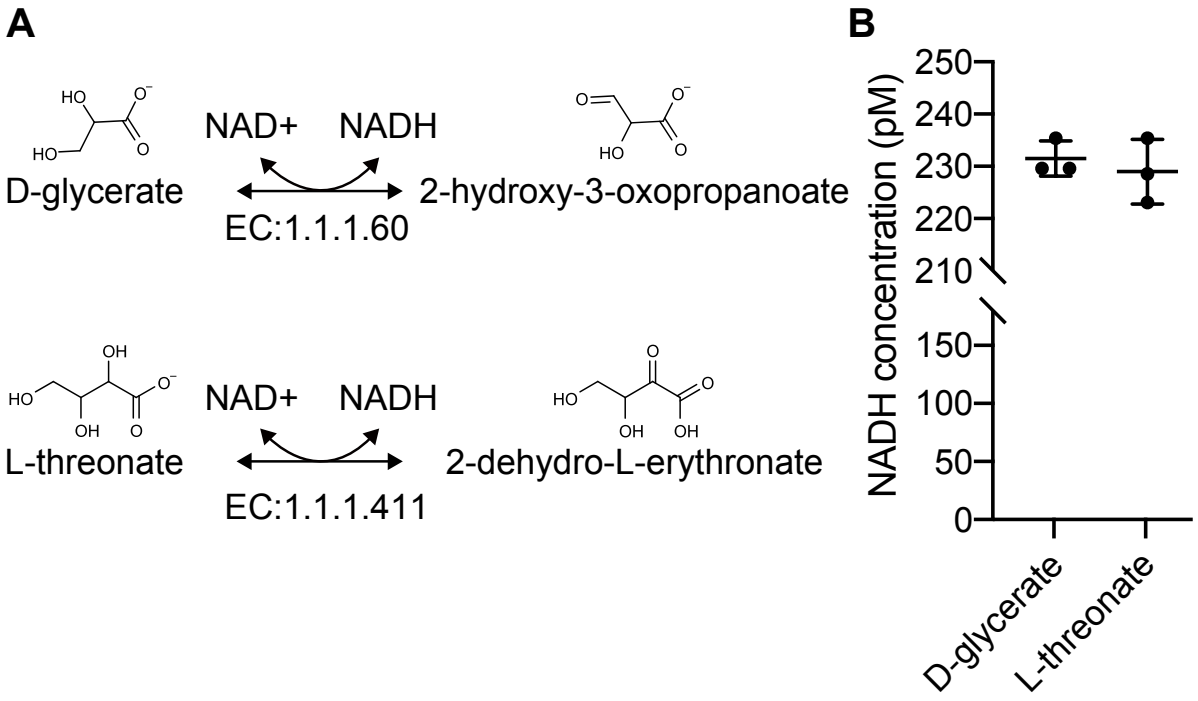

Fig. S7. In vitro enzyme assay results for a putative L-threonate dehydrogenase encoded by *E. coli ygbJ*. (*A*) Two proposed metabolic reactions for YgbJ based on the EC number 1.1.1.60 predicted by DeepEC (above) and EC number 1.1.1.411 available at UniProt database (below). (*B*) Results of the in vitro enzyme assays using D-glycerate (1.1.1.60) and L-threonate (1.1.1.411) as substrates. Enzymatic activities for D-glycerate and L-threonate were determined by measuring the total NADH level (pM) using the NAD/NADH quantitation colorimetric kit. The resulting NADH concentrations for the two reactions involving D-glycerate and L-threonate were estimated based on the standard curve (see *SI Appendix, Materials and Methods*). The enzyme assays were conducted in triplicates.

SI Appendix Tables

Table S1. Precision and recall values for the prediction of EC numbers for potential promiscuous enzymes (from 43,124 protein sequences having two or more EC numbers in the gold standard dataset).

Number of EC numbers per protein sequence (number of corresponding protein sequences)	Precision	Recall
2 (36,413)	0.940	0.905
3 (1,251)	0.825	0.709
4 (396)	0.775	0.720
5 (212)	0.629	0.385

Table S2. Prediction performances of six different, locally installable EC number prediction tools using 2,310 enzyme protein sequences as inputs, which were previously considered for the development of all these tools (from the Swiss-Prot database released March 2007).

EC number prediction tool	Precision	Recall	Run time (sec)
DeepEC	0.993	0.946	66
CatFam	0.991	0.963	450
DETECT v2	0.980	0.935	65,566
ECPred	0.990	0.952	344,880
EFICAz2.5	0.949	0.913	411,024
PRIAM	0.979	0.954	321

SI Appendix References

1. Bairoch A & Apweiler R (2000) The SWISS-PROT protein sequence database and its supplement TrEMBL in 2000. *Nucleic Acids Res.* 28(1):45-48.
2. UniProt C (2015) UniProt: a hub for protein information. *Nucleic Acids Res.* 43(Database issue):D204-212.
3. Kumar N & Skolnick J (2012) EFICAz2.5: application of a high-precision enzyme function predictor to 396 proteomes. *Bioinformatics* 28(20):2687-2688.
4. Seo S, Oh M, Park Y, & Kim S (2018) DeepFam: deep learning based alignment-free method for protein family modeling and prediction. *Bioinformatics* 34(13):i254-i262.
5. Zeng H, Edwards MD, Liu G, & Gifford DK (2016) Convolutional neural network architectures for predicting DNA-protein binding. *Bioinformatics* 32(12):i121-i127.
6. Kim HK*, et al.* (2018) Deep learning improves prediction of CRISPR-Cpf1 guide RNA activity. *Nat. Biotechnol.* 36(3):239-241.
7. Alipanahi B, Delong A, Weirauch MT, & Frey BJ (2015) Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning. *Nat. Biotechnol.* 33(8):831-838.
8. Buchfink B, Xie C, & Huson DH (2015) Fast and sensitive protein alignment using DIAMOND. *Nat. Methods* 12(1):59-60.
9. Ioffe S & Szegedy C (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. *Proc. 32nd Int. Conference on Machine Learning*:448–456.
10. Kingma DP & Ba J (2014) Adam: a method for stochastic optimization. *arXiv*.
11. Li Y*, et al.* (2018) DEEPre: sequence-based enzyme EC number prediction by deep learning. *Bioinformatics* 34(5):760-769.
12. Finn RD*, et al.* (2014) Pfam: the protein families database. *Nucleic Acids Res.* 42(Database issue):D222-230.
13. Morrison KL & Weiss GA (2001) Combinatorial alanine-scanning. *Curr. Opin. Chem. Biol.* 5(3):302-307.
14. Rose PW*, et al.* (2017) The RCSB protein data bank: integrative view of protein, gene and 3D structural information. *Nucleic Acids Res.* 45(D1):D271-D281.
15. Bakan A*, et al.* (2014) Evol and ProDy for bridging protein sequence evolution and structural dynamics. *Bioinformatics* 30(18):2681-2683.
16. Abadi M*, et al.* (2016) TensorFlow: large-scale machine learning on heterogeneous distributed systems. *arXiv*:1603.04467.
17. Pedregosa F*, et al.* (2011) Scikit-learn: machine learning in python. *J. Mach. Learn. Res.* 12:2825-2830.