finding genes

unannotated (or poorly annotated) sequences:

- novel genomes
- novel genes (sequenced from RNA)

‘genes’ are not just transcribed (protein coding) DNA

- translated and untranslated regions
- *cis* regulatory regions (hardest to find)

currently, we can only find transcribed DNA (easily)

- (1) similarity to annotated sequences
- (2) predict based on common features

finding genes: similarity…

similarity search using tools such as BLAST

- depends on quality of the database annotation (experimental vs. inferred)
- hits may not be very useful
  - e.g. anonymous expressed sequences
  - i.e. refSeq is a better than all of GenBank

depends on sequence similarity

- distant relatives of model systems do not usually work (well)
- intragenome comparisons may not work well
  - most genes are members of large families

finding genes: …similarity…

(near) exact nucleotide or amino acid match

=> adopt database annotation

distant match

=> provisionally adopt database annotation

complex patterns

e.g. match–mismatch–match

could be exon–intron–exon (or many other things)

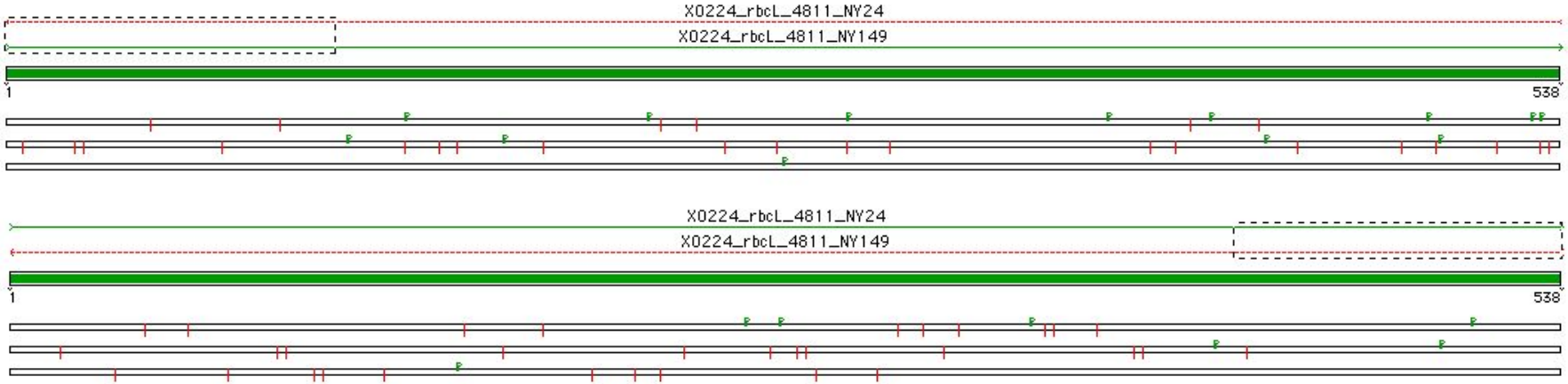
e.g. match–mismatch

could be exon–spacer or novel gene, etc.

finding genes: …similarity

(usually) need to dismember sequences for more informative matches

- logical portions (e.g. reads, clones, contigs, etc.)
  - may not be biologically meaningful
- small (ca. 500–5000 bp) sliding window
  - not biologically meaningful
  - difficult to resolve conflicting annotations
- open reading frames
  - more difficult to identify than one might think


XO224_rbcL_4811_NY24
XO224_rbcL_4811_NY149
1
538
XO224_rbcL_4811_NY24
XO224_rbcL_4811_NY149
1
538

finding genes: common features…

often called ‘*ab initio*’ (from the beginning)

look for long open reading frames (ORF)

- assumes ‘universal’ (standard) genetic code
  - unknown how non–standard the code actually is
- identification of splice sites key to finding ‘real’ ORF
  - often GUAG (spliceosome) with many variations
  - RNA editing makes this difficult
  - remapping RNA sequence data is often the most effective
- cannot identify ‘different’ exons as the variation is unknown

finding genes: …common features…

the path of fewer assumptions:

- A/T vs. G/C content
- kmer frequency (often 6–mers)
- use DFA, neural networks, SVM, Markov models, etc.
  - require a ‘training set’ of known genes
  - still cannot find ‘different’ exons
- unsupervised training
  - requires a ‘training set’ of unannotated sequence

finding genes: …common features

k–order Hidden Markov Models

- probability of {A|C|G|T} given previous *k* bases
- *k* usually equals 5
  - i.e. the frequency of 6–mers

multiple models can be used simultaneously

- e.g. one for each codon position

for speed, interpolated Markov models (IMM) are used

- averages several models into one
- (could have just used a simpler/different model)

finding genes: software (plant biased)...

non–vigorous and non–uniform benchmarks

- often programs work well on A and fail on B
- (i.e. they have over–parameterized models)

GENSCAN (Burge & Karlin 1997; Burge 1998)

- very model dependent (i.e. not so useful)
- no longer under active development
- executables are free for academic use

finding genes: …software...

EuGène (Schiex et al. 2001)

- under active development (mostly for prokaryotes)
- additional types of models added via plugins
- additional features can be added via plugins
- open source

geneid (Blanco et al. 2002)

- (no longer) under active development
- user can specify models (annoying/tedious format)
- open source

finding genes: …software...

GeneMark (Ter–Hovhannisyan et al. 2008)

- under active development
- many different ‘flavors’ are available
- GeneMark.hmm ES
  - self–training, hidden Markov models
  - requires ca. 10 Mbp of sequence to make model
  - very useful for non–model species
- executables are free for academic use

finding genes: …software...

AUGUSTUS (Stanke and Waack 2003)

- under active development
- uses a Hidden–Markov Model (HMM)
  - G/C content (4 classes), intron length, splice site, and splice site adjacent positions
- a plant/algae/fungi trained version is available
- freely available

finding genes: …software

BRAKER (Hoff et al. 2016) BREAKER2 (Brůna et al. 2021)

- GeneMark-ES/EP+ plus AUGUSTUS
- predict genes *ab initio* with GeneMark-ES
- produce augmented data
  - find similar coding sequences using DIAMOND
  - combine DIAMOND hits with genome background via Spaln
- train GeneMark-EP+ with original plus augmented data
- use GeneMark-EP+ output to train AUGUSTUS
- predict genes with AUGUSTUS

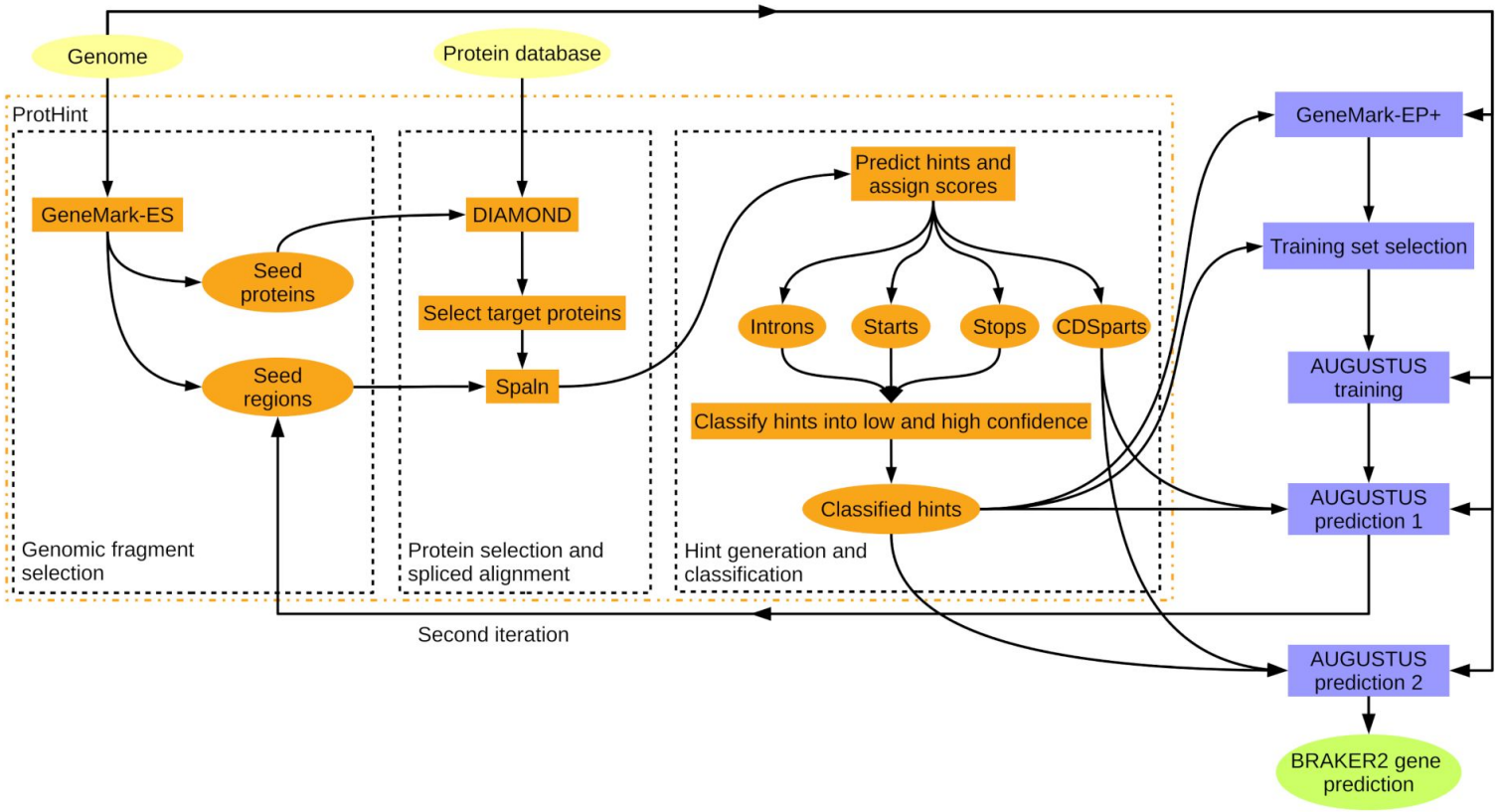
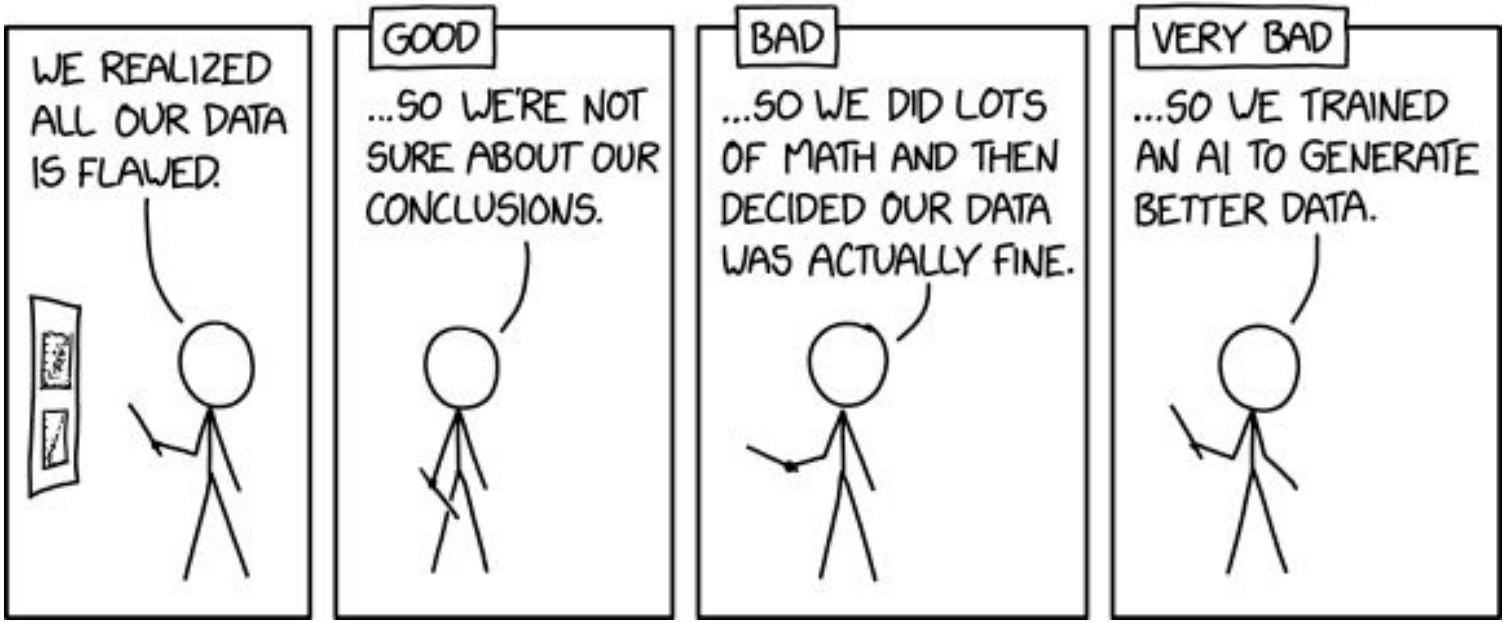


Figure 1. Flowchart of the BRAKER2 pipeline. Input, intermediate and output data are shown by ovals. The tools and processes of the ProtHint pipeline are shown in orange; other components of BRAKER2 are shown in blue.

(Brůna et al. 2021; https://doi.org/10.1093/nargab/lqaa108)



https://xkcd.com/2494

gene function: Gene Ontology (GO)...

a controlled vocabulary for genes (Ashburner et al. 2000)

- cellular/extracellular localization (C or CC or CCO)
- biochemical function (F or MF or MFO)
- type of biological processes (B or BP or BPO)

designed to be applicable to all of life

each term has a unique identifier

related terms are grouped together into broader categories

- directed acyclic graph
- http://www.geneontology.org

gene function: …Gene Ontology (GO)...

ribulose-1,5-bisphosphate carboxylase/oxygenase

GO:0015977: carbon fixation

GO:0009573: chloroplast

GO:0009536: plastid

gene function: …Gene Ontology (GO)...

GO:0015977: carbon fixation

ancestors:

is_a (inferred) biological_process

is_a (inferred) metabolic process

is_a (inferred) single-organism metabolic process

is_a organic substance metabolic process

children:

C4 photosynthesis is_a carbon fixation

CAM photosynthesis is_a carbon fixation

carbon fixation by 3-hydroxypropionate cycle is_a carbon fixation

carbon fixation by acetyl-CoA pathway is_a carbon fixation

reductive pentose-phosphate cycle part_of carbon fixation

reductive tricarboxylic acid cycle is_a carbon fixation

gene function: …Gene Ontology (GO)...

full database can be downloaded

database structure is not optimized for speed

- modify for serious analytic use
  - delete unused/redundant columns/tables
  - create indices and foreign keys as needed
  - remove unused indices

NCBI files also needed (gene_info, gene2accession)

BLAST (or similar) against GenBank (full or subset)

query BLAST hits against GO database

gene function: …Gene Ontology (GO)...

BLAST2GO (Conesa et al. 2005)

a (JAVA) gui to BLAST and query the GO database

uses the ‘stock’ GO database => slow

PANNZER (Koskinen et al. 2015)

BLAST + regression = GO term

DEEPred (Rifaioglu et al. 2019)

a collection of small DNNs which each predict 4–5 GO terms

DNNs are organized in a hierarchy that follows the GO graph

sequences are input as: 3mer frequency (7–letter amino acid alphabet), amino acid sequence composition (50 physicochemical properties; PAAC), and frequency of clustered kmers (subsequence profile map; SPMap)

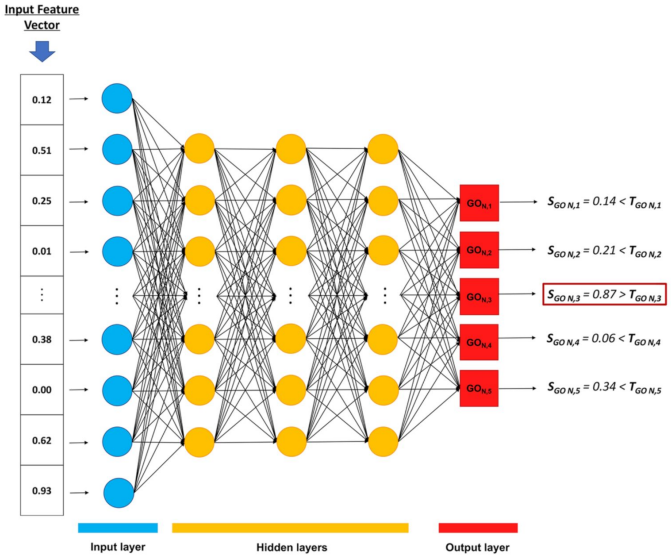


Figure 3. The representation of an individual multi-task feed-forward DNN model of DEEPred (i.e., model N). Here, each task at the output layer (i.e., red squares) corresponds to a different GO term. In the example above, a query input vector is fed to the trained model N and a score greater than the pre-defined threshold is produced for GON,3, which is marked as a prediction.

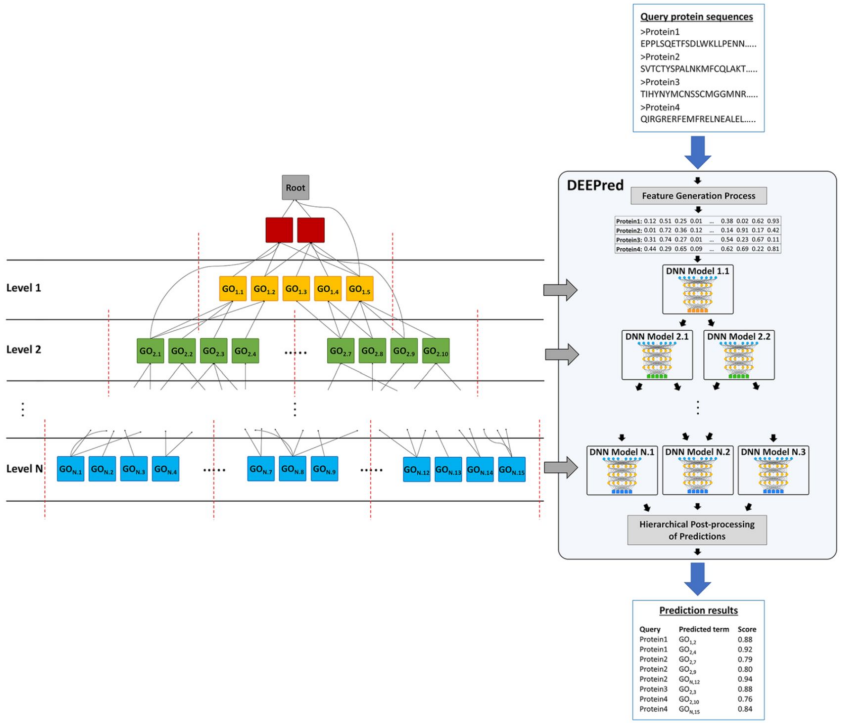


Figure 4. Illustration of the GO-level-based architecture of DEEPred on a simplified hypothetical GO DAG. We omitted highly generic GO terms (shown with red colored boxes) at the top of the GO hierarchy (e.g., GO:0005488 - Binding) from our models, since they are less informative and their training datasets are highly heterogeneous. In the illustration, DNN model 1.1 incorporates GO terms: $GO_{1,1}$ to $GO_{1,5}$ from GO-level 1. In the real application, most of the GO levels were too crowded to be modeled in one DNN; in these cases, multiple DNN models were created for the same GO level (red dashed lines represent how GO terms are grouped to be modeled together). In this example, DNN models N.1, N.2 and N.3 incorporates GO terms: $GO_{N,1}$ to $GO_{N,5}$, $GO_{N,6}$ to $GO_{N,10}$, $GO_{N,11}$ to $GO_{N,15}$; respectively, due to the high number of GO terms on level N. At the prediction step, when a list of query sequences is run on DEEPred, all sequences are transformed into feature vectors and fed to the multi-task DNN models. Afterwards, GO term predictions from each model are evaluated together in the hierarchical post-processing procedure to present the finalized prediction list.

(Rifaioglu et al. 2019; https://doi.org/10.1038/s41598-019-43708-3)

gene function: …Gene Ontology (GO)...

SDN2GO (Cai et al. 2020)

- three conjoined neural networks
- input data are 3mer frequency, predicted protein–protein interactions, domain presence/absence
- restricted to well–studied model organisms

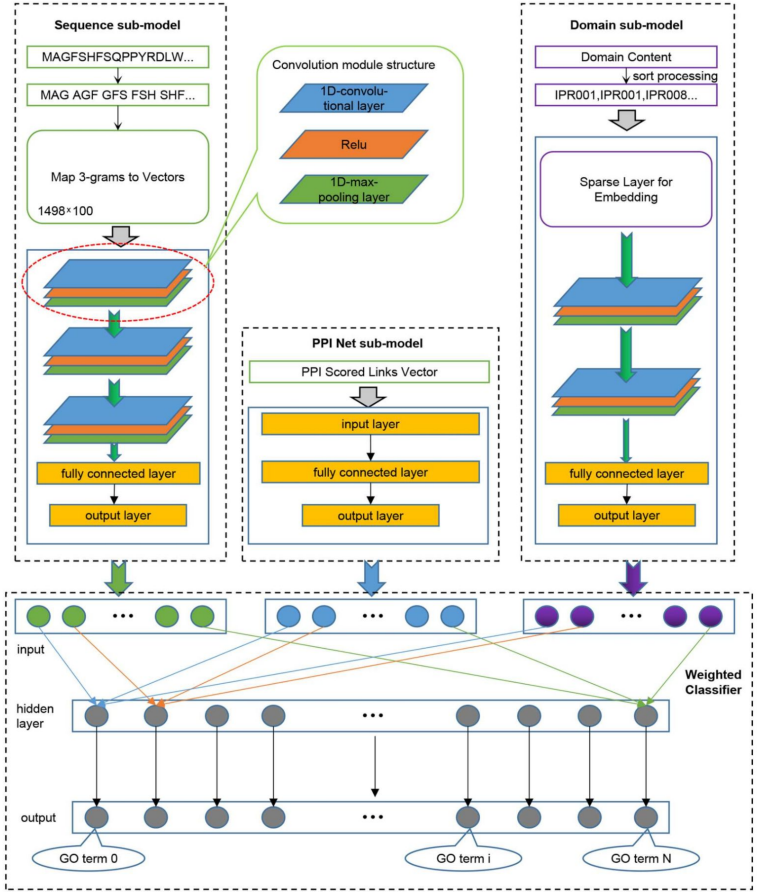


FIGURE 1 | The integrated deep learning model architecture. (1) The Sequence sub-model utilizes 1-Dimensional convolutional neural networks to extract features from sequence input, which was encoded as 3-grams and then mapped to 3-grams-vector-matrix. (2) The PPI Net sub-model is generated to dense the features from PPI Network using classical neural networks. (3) The Domain sub-model initializes a Sparse layer, which is integrated into the sub-model to optimize, to generate a lookup table for domains, and the sorted domains sentence processed by the Sparse layer is entered into 1-Dimensional convolutional neural networks to extract features. (4) All the output features of the three sub-models are combined and entered into the Weighted Classifier, and the output vector represents the probability of GO terms.

TABLE 2 | The comparison results of the competing method on the independent testing set.

Method	BP			MF			CC		
	F_{max}	AUPR	AUC	F_{max}	AUPR	AUC	F_{max}	AUPR	AUC
BLAST	0.347	0.192	0.771	0.381	0.292	0.873	0.386	0.245	0.860
DeepGO	0.321	0.095	0.729	0.291	0.117	0.784	0.210	0.080	0.687
NetGO	0.173	0.048	0.594	0.386	0.243	0.919	0.217	0.092	0.669
SN2GO	0.132	0.044	0.893	0.423	0.306	0.953	0.384	0.264	**0.948**
SDN2GO	**0.361**	**0.203**	**0.917**	**0.561**	**0.471**	**0.964**	**0.432**	**0.290**	0.947

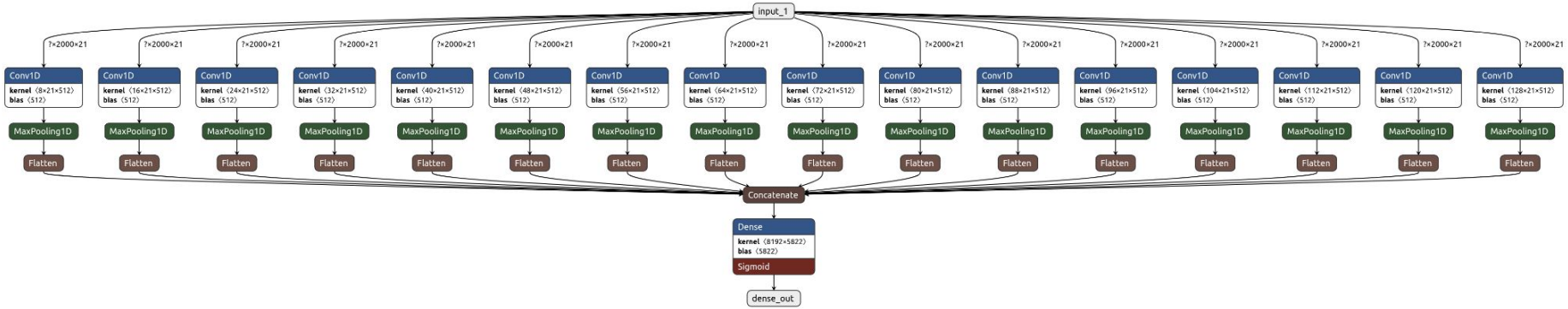
The bold values indicate the best values.

(Cai et al. 2020; https://doi.org/10.3389/fbioe.2020.00391)

gene function: …Gene Ontology (GO)...

DeepGOPlus (Kulmanov and Hoehndorf 2021)

- improved DeepGO (Kulmanov et al. 2018)
- Convolutional Neural Network (CNN)
- input data are one–hot encoded amino acid sequence
- model predictions are weighted by DIAMOND matches



(Kulmanov and Hoehndorf 2021; https://doi.org/10.1093/bioinformatics/btaa763)

Table 3. The comparison of performance on the second dataset generated by a time-based split

Method	F_{max}			S_{min}			AUPR		
	MFO	BPO	CCO	MFO	BPO	CCO	MFO	BPO	CCO
Naive	0.306	0.318	0.605	12.105	38.890	9.646	0.150	0.219	0.512
DiamondBLAST	0.525	0.436	0.591	9.291	39.544	8.721	0.101	0.070	0.089
DiamondScore	0.548	0.439	0.621	8.736	34.060	7.997	0.362	0.240	0.363
DeepGO	0.449	0.398	0.667	10.722	35.085	7.861	0.409	0.328	0.696
DeepGOCNN	0.409	0.383	0.663	11.296	36.451	8.642	0.350	0.316	0.688
DeepText2GO	**0.627**	0.441	0.694	5.240	17.713	**4.531**	**0.605**	0.336	**0.729**
GOLabeler	0.580	0.370	0.687	**5.077**	**15.177**	5.518	0.546	0.225	0.700
DeepGOPlus	0.585	**0.474**	**0.699**	8.824	33.576	7.693	0.536	**0.407**	0.726

(Kulmanov and Hoehndorf 2021; https://doi.org/10.1093/bioinformatics/btaa763)

gene function: …Gene Ontology (GO)

compare GO terms among samples

shared versus unique

(use vague [ancestral] terms)

make candidate lists based on statistical increase/decrease in frequency of GO terms

gene function: InterPro...

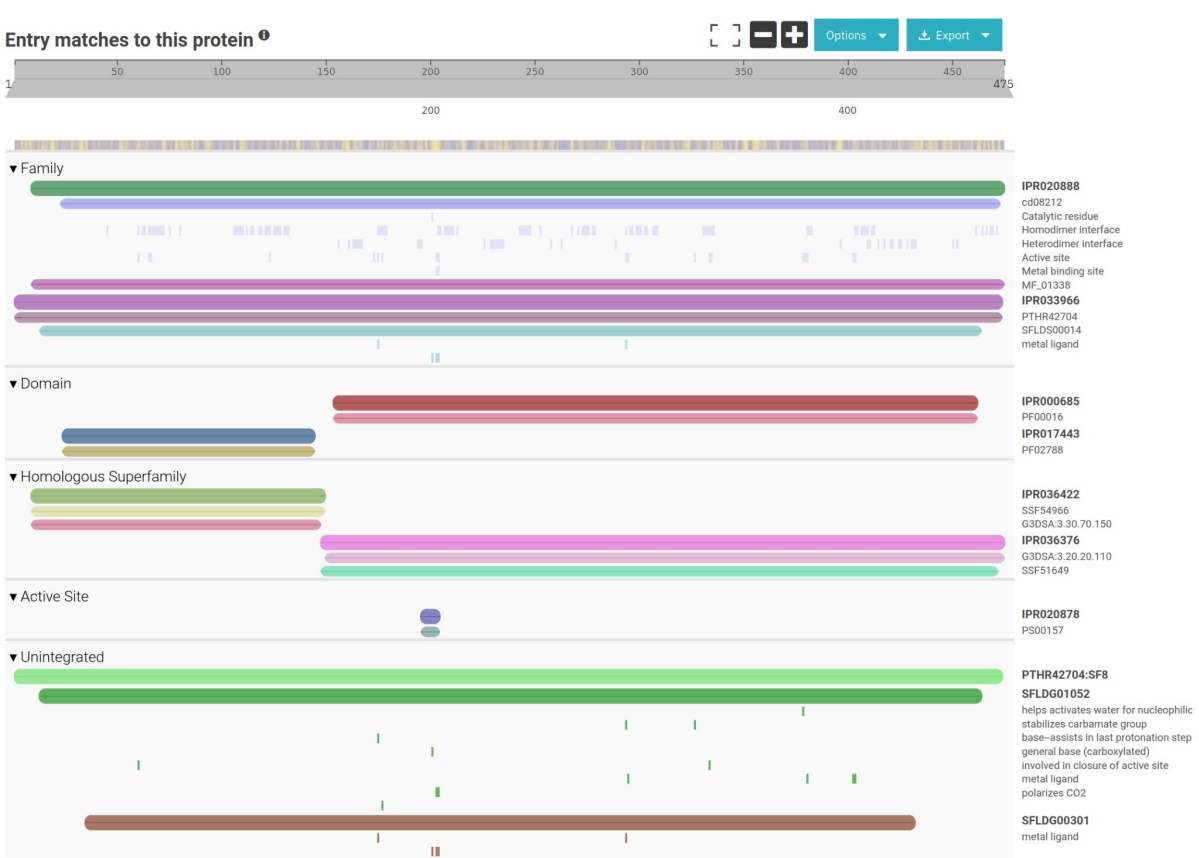
a consortium of 13 protein annotation databases
- each database has a different focus and style
- not a comprehensive atlas of protein function

a collection of ‘signatures’ for annotations
- functional as well as [super|sub]familial classification
  - uses a mixture of homology definitions
  - categories are not mutually exclusive
- models for various motifs (‘signatures’)
  - e.g. Hidden Markov, regular expressions

gene function: ...InterPro

InterProScan runs all models on input sequences

- a mixture of Java, Perl, Python, etc.
- 64-bit Linux standalone or (limited) web
  - http://www.ebi.ac.uk/interpro/search/sequence/
- outputs possible functional annotations
- often GO terms are associated with annotations



GO terms

Biological Process

- carbon fixation (GO:0015977)

Molecular Function

- ribulose-bisphosphate carboxylase activity (GO:0016984)
- magnesium ion binding (GO:0000287)

Cellular Component

None

And now for something completely different…

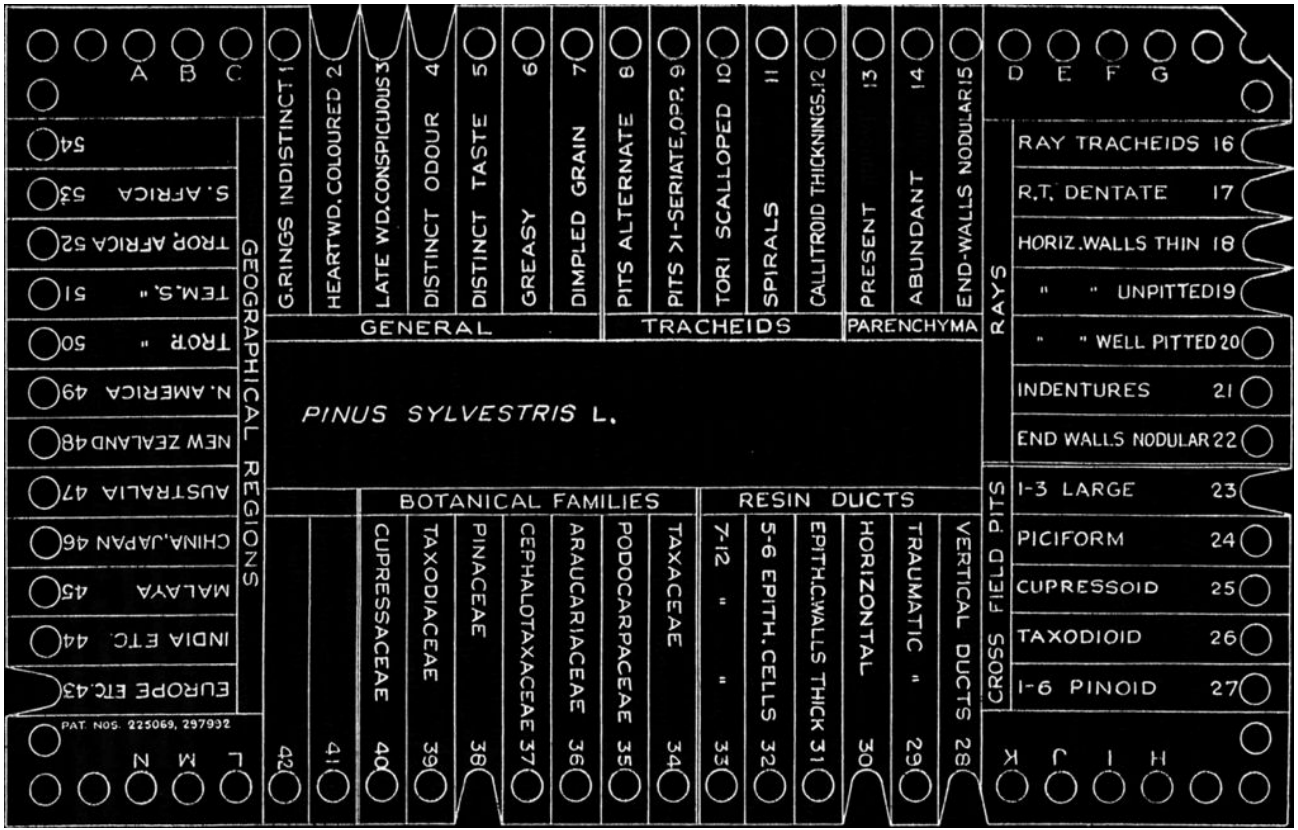
databases

tools for (electronic) information storage and retrieval

- analogue: reference books, herbarium, card index
- electronic: text files, spreadsheets, specialized programs

the heart of bioinformatics

- complex questions require well–designed and curated databases
- consistent data representation
- highly atomized data
- (fully) descriptive of the studied phenomenon



Phillips, E. W. J. 1941. The identification of coniferous woods by their microscopic structure. The Journal of the Linnean Society of London, Botany 52: 259–320.

databases: types

simple table (aka flat file)

columns specifying fields, one row for each record

hash (distributed or local)

(two column) simple table(s); unique key for each value

object oriented

each record stored as an object (sensu programming)

relational table

many simple tables with common fields linked among tables

databases: simple table...

familiar data structure: one item/event/concept and its attributes (e.g. a spreadsheet)

analyses are done directly on the data file

- sort, find, sum, count, etc.

often there are repeated data elements

- errors are more common
- effort is required to standardize and correct data
- storage inefficient

cannot (easily) represent all data

- inapplicable, missing, polymorphic entries are problematic

databases: ...simple table

	attribute 0	attribute 1	attribute 2
record 0	TRUE	9.5	*Cupressus*
record 1	TRUE	9.2	*Cupressus*
record 2	TRUE	9.1	*Thujopsis*
record 3	FALSE	9.6	*Thuja*

databases: hash...

(distributed) key/value stores

usually very fast computationally (all data held in RAM)

works with big data (when distribute to multiple computers)

can only (efficiently) represent some relationships:

e.g. x => y; y => z; but not z => y

therefore, queries must be defined before building the database

used for large web databases (e.g. Facebook, Google)

open source highlights: F14, Kademlia, Voldemort, Cassandra

databases: ...hash...

	attribute 0
record 0	TRUE
record 1	TRUE
record 2	TRUE
record 3	FALSE

	attribute 1
record 0	9.5
record 1	9.2
record 2	9.1
record 3	9.6

	attribute 2
record 0	*Cupressus*
record 1	*Cupressus*
record 2	*Thujopsis*
record 3	*Thuja*

databases: ...hash

	attribute 0 | attribute 1 | attribute 2
record 0	TRUE | 9.5 | *Cupressus*
record 1	TRUE | 9.2 | *Cupressus*
record 2	TRUE | 9.1 | *Thujopsis*
record 3	FALSE | 9.6 | *Thuja*

databases: object oriented...

programming ‘objects’

- defined properties and data types (fields)
- can have many of the same type of object
  - populated with different values

can represent any data/relationship

- ‘faster’ for many types of searches
- some use SQL or ‘SQL–like’ queries

open source highlights: C2, Perst, Texpress

databases: ...object oriented

```
"record 0": {
    "attribute 0":TRUE
    "attribute 1":9.5
    "attribute 2":"Cupressus"
}

  "record 1": {
      "attribute 0":TRUE
      "attribute 1":9.2
      "attribute 2":"Cupressus"
  }
```

```
"record 2": {
    "attribute 0":TRUE
    "attribute 1":9.1
    "attribute 2":"Thujopsis"
}

  "record 3": {
      "attribute 0":FALSE
      "attribute 1":9.6
      "attribute 2":"Thuja"
  }
```

databases: relational table...

many simple tables with common fields linked among tables

use unique ‘ID’ fields to relate data elements

not (usually) seen in the final output

linear relationships (one–to–one)

x <=> y <=> z

x => y => z

x <=> y => z

databases: ...relational table...

polymorphic relationships (one–to–many)

x => y || z

hierarchical relationships (one–to–many)

((((x y) (z a)) b) c)

complex networks (many–to–many)

((x y) z) <=> (a (b c)); (x y z) <=> (a b c)

databases: ...relational table...

almost all analyses require a query

- queries construct temporary simple tables with specific purpose(s)
- rows possibly repeated (e.g. polymorphism)

stored table structure has (almost) no repeated data elements

- errors are less frequent
- easy to standardize and correct data
- data are ‘compressed’

can represent any pattern of relationship

- some queries may difficult for some data structures

databases: ...relational table...

Codd, E. F. 1970. A relational model of data for large shared data banks. Communications of the ACM [Association for Computing Machinery] 13: 377–387.

System R (IBM)

introduced SQL (Structured Query Language)

first sold to Pratt & Whitney (1977)

proprietary highlights:

DB2 (IBM), MaxDB (SAP), Oracle (Oracle), SQL Server (Microsoft)

open source highlights: MySQL/MariaDB, PostgreSQL, SQLite

databases: ...relational table...

ID	attribute 0
0	TRUE
1	FALSE

ID	attribute 1
0	9.5
1	9.2
2	9.1
3	9.6

ID	attribute 2
0	*Cupressus*
1	*Thujopsis*
2	*Thuja*

databases: ...relational table...

ID	attribute 0	attribute 2
0	0	0
1	0	0
2	0	1
3	1	2

ID	attribute 1	attribute 2
0	0	0
1	1	0
2	2	1
3	3	2

simple table

given name(s)	family name	born	died	abbreviation	interest(s)	alias(es)
George	Bentham	1800	1884	Benth.	Mycology; Pteridophytes; Spermatophytes	
Ezra	Brainerd	1844	1924	Brainerd	Spermatophytes	
Nathaniel Lord	Britton	1859	1934	Britton	Bryophytes; Mycology; Pteridophytes; Spermatophytes	
Merritt Lyndon	Fernald	1873	1950	Fernald	Pteridophytes; Spermatophytes	
Asa	Gray	1810	1888	A.Gray	Algae; Bryophytes; Pteridophytes; Spermatophytes	
Oskar Eric Gunnar	Hultén	1894	1981	Hultén	Bryophytes; Pteridophytes; Spermatophytes	
Carl	Linnaeus	1707	1778	L.	Algae; Bryophytes; Mycology; Pteridophytes; Spermatophytes	Linné, Carl von

hash 0

abbreviation	given name(s)
Benth.	George
Brainerd	Ezra
Britton	Nathaniel Lord
Fernald	Merritt Lyndon
A.Gray	Asa
Hultén	Oskar Eric Gunnar
L.	Carl

hash 1

abbreviation	family name
Benth.	Bentham
Brainerd	Brainerd
Britton	Britton
Fernald	Fernald
A.Gray	Gray
Hultén	Hultén
L.	Linnaeus

hash 2

abbreviation	born
Benth.	1800
Brainerd	1844
Britton	1859
Fernald	1873
A.Gray	1810
Hultén	1894
L.	1707

hash 3

abbreviation	died
Benth.	1884
Brainerd	1924
Britton	1934
Fernald	1950
A.Gray	1888
Hultén	1981
L.	1778

hash 4

abbreviation	interest(s)
Benth.	Mycology; Pteridophytes; Spermatophytes
Brainerd	Spermatophytes
Britton	Bryophytes; Mycology; Pteridophytes; Spermatophytes
Fernald	Pteridophytes; Spermatophytes
A.Gray	Algae; Bryophytes; Pteridophytes; Spermatophytes
Hultén	Bryophytes; Pteridophytes; Spermatophytes
L.	Algae; Bryophytes; Mycology; Pteridophytes; Spermatophytes

hash 5

abbreviation	alias(es)
Benth.	
Brainerd	
Britton	
Fernald	
A.Gray	
Hultén	
L.	Linné, Carl von

Benth.

attribute	value
given name(s)	George
family name	Bentham
born	1800
died	1884
interest(s)	Mycology; Pteridophytes; Spermatophytes

A.Gray

attribute	value
given name(s)	Asa
family name	Gray
born	1810
died	1888
interest(s)	Algae; Bryophytes; Pteridophytes; Spermatophytes

Brainerd

attribute	value
given name(s)	Ezra
family name	Brainerd
born	1844
died	1924
interest(s)	Spermatophytes

Hultén

attribute	value
given name(s)	Oskar Eric Gunnar
family name	Hultén
born	1894
died	1981
interest(s)	Bryophytes; Pteridophytes; Spermatophytes

Britton

attribute	value
given name(s)	Nathaniel Lord
family name	Britton
born	1859
died	1934
interest(s)	Bryophytes; Mycology; Pteridophytes; Spermatophytes

L.

attribute	value
given name(s)	Carl
family name	Linnaeus
born	1707
died	1778
interest(s)	Algae; Bryophytes; Mycology; Pteridophytes; Spermatophytes
alias(es)	Linné, Carl von

Fernald

attribute	value
given name(s)	Merritt Lyndon
family name	Fernald
born	1873
died	1950
interest(s)	Pteridophytes; Spermatophytes

botanists

botanist ID	abbreviation	born	died
1	Benth.	1800	1884
2	Brainerd	1844	1924
3	Britton	1859	1934
4	Fernald	1873	1950
5	A.Gray	1810	1888
6	Hultén	1894	1981
7	L.	1707	1778

names

name ID	botanist ID	given name(s)	family name
1	1	George	Bentham
2	2	Ezra	Brainerd
3	3	Nathaniel Lord	Britton
4	4	Merritt Lyndon	Fernald
5	5	Asa	Gray
6	6	Oskar Eric Gunnar	Hultén
7	7	Carl	Linnaeus
8	7	Carl von	Linné

interests

interest ID	interest
1	Algae
2	Bryophytes
3	Mycology
4	Pteridophytes
5	Spermatophytes

botanical interests

botanical interest ID	botanist ID	interest ID
1	1	3
2	1	4
3	1	5
4	2	5
5	3	2
6	3	3
7	3	4
8	3	5
9	4	4
10	4	5
11	5	1
12	5	2
13	5	4
14	5	5
15	6	2
16	6	4
17	6	5
18	7	1
19	7	2
20	7	3
21	7	4
22	7	5

Structured Query Language...

```
SELECT
  CASE (bottlecount)
    WHEN 0 THEN 'No more bottle of beer on the wall, no more bottles of beer. ' ||
                'Go to the store and buy some more, 99 bottles of beer on the wall.'
    WHEN 1 THEN '1 bottle of beer on the wall, 1 bottle of beer. ' ||
                'Take one down and pass it around, no more bottles of beer on the wall.'
    WHEN 2 THEN '2 bottles of beer on the wall, 2 bottles of beer. ' ||
                'Take one down and pass it around, 1 bottle of beer on the wall.'
    ELSE
      rtrim (cast((BottleCount) as char(2))) || ' bottles of beer on the wall, ' ||
      rtrim (cast((BottleCount) as char(2))) || ' bottles of beer. ' ||
      'Take one down and pass it around, ' ||
      rtrim (cast((BottleCount)-1 as char(2))) || ' bottles of beer on the wall.'
  END
FROM
(
  SELECT avalue * 10 + bvalue as bottlecount
  FROM
    (VALUES (9), (8), (7), (6), (5), (4), (3), (2), (1), (0)) a(avalue),
    (VALUES (9), (8), (7), (6), (5), (4), (3), (2), (1), (0)) b(bvalue)
) as valuelist;
```

...Structured Query Language...

based on work by Codd (1970)

first put into practice (at least) twice:

- D.D. Chamberlin and R.F. Boyce: System R (IBM)
  - eventually called SQL
  - System R is now SQL/DS and DB2
- M. Stonebraker and E. Wong: Ingres (UC Berkeley)
  - QUEL was very similar to SQL
  - Ingres is now Ingres, Sybase, SQL Server, PostgreSQL, etc.

...Structured Query Language

an (inconsistently implemented) international standard

- SQL-92, -1999, -2003, -2006, -2008, -2011, -2016
- often ‘extended’ to allow non–standard syntax
- or to easily handle ‘unusual’ data types

strongly typed (weakened by some implementations)

keywords written in ALL CAPS (for readability only)

lines end with semicolon;

unnecessarily complex (takes after COBOL)

MariaDB...

MySQL started in 1994 by M. Widenius and D. Axmark

2008 Sun Microsystems (now Oracle) buys MySQL

2008 MariaDB forked from MySQL

designed to be a ‘drop in’ replacement for MySQL

connectors/libraries for MySQL (should) work with MariaDB

features are slowly diverging from MySQL

the ‘default’ open source relational database

PostgreSQL and SQLite are also common (but not as scaleable)

...MariaDB

a flexible client/server framework

both client and server can be installed on one computer

server can be installed on multiple computers (cluster)

supports numerous ‘locals’

a single connection client for multiple database types

a single SQL interpreter for multiple storage engines

storage engines assigned to each table