On computable protein functions

Harry Scholes

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
of
UCL.

Institute of Structural and Molecular Biology
UCL

November 28, 2020

I, Harry Scholes, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

Abstract

Proteins are biological machines that perform the majority of functions necessary for life. Nature has evolved many different proteins, each of which perform a subset of an organism's functional repertoire. One aim of biology is to solve the sparse high dimensional problem of annotating all proteins with their true functions. Experimental characterisation remains the gold standard for assigning function, but is a major bottleneck due to resource scarcity. In this thesis, we develop a variety of computational methods to predict protein function, reduce the functional search space for proteins, and guide the design of experimental studies. Our methods take two distinct approaches: protein-centric methods that predict the functions of a given protein, and function-centric methods that predict which proteins perform a given function. We applied our methods to help solve a number of open problems in biology. First, we identified new proteins involved in the progression of Alzheimer's disease using proteomics data of brains from a fly model of the disease. Second, we predicted novel plastic hydrolase enzymes in a large data set of 1.1 billion protein sequences from metagenomes. Finally, we optimised a neural network method that extracts a small number of informative features from protein networks, which we used to predict functions of fission yeast proteins.

Impact statement

We present a variety of computational methods to predict the chemical and biological functions that proteins perform.

First, we characterise how the brain proteome changes during progression of Alzheimer's disease, by monitoring which proteins are affected and how their abundances change. Despite vast expenditure over many decades, the molecular mechanisms of Alzheimer's disease, and effective treatments against it, remain elusive. Knowledge of which proteins and pathways are affected by Alzheimer's disease are required to develop drugs against the disease. Disease severity is, in part, determined by the patient genotype, so characterising proteome changes associated with different types of Alzheimer's disease will aid the development of precise and personalised approaches to treatment. This may be especially relevant to early-stage interventions to prevent Alzheimer's disease, or improve quality of life, in individuals with susceptible genetic backgrounds. Given the sluggish progress of the field, it would be imprudent to pass comment on time frames for these developments, but suffice it to say that they will be lucrative.

Second, we mined metagenomes for putative plastic hydrolase enzymes that break plastic polymers down into their constiuent monomers. Virgin plastics are produced from non-renewable petroleum sources, with a large carbon footprint. Despite efforts to increase the amount that is recycled, only 14% of the plastic produced each year is *collected* for recycling. This is largely because the process is uneconomic, but also because recycled plastics have inferior properties. In the future, plastic hydrolases may form the fulcrum of a profitable, circular plastic economy, where high-quality recycled plastics are regenerated from used plastics in an energy efficient process. Currently, many unrecycled plastics go to landfill, but some end up as pollutants, either by being incinerated, which further increases the carbon footprint, or by polluting the environment directly. Plastic hydrolases, and the bacteria that use these enzymes to metabolise plastic, may, one day, help to clean up the

environmental plastic pollution that affects every continent and ocean on Earth. Taken together, plastic hydrolases may have a profound impact on the health of organisms and the environment globally.

Finally, we predicted the functions of proteins from a species of yeast that is an important model for understanding cellular processes in higher eukaryotes, including humans. Predicted functions help guide the design of experimental studies at the microscopic level, to confirm or refute predicted functions; at the mesoscopic level, to disentangle the interplay between proteins and pathways to determine their cellular effects; and at the macroscopic level, to identify the causes of diseases and how to treat or prevent them. Beyond basic science, predicted functions have a broad range of commercial applications, including, but not limited to, synthetic biology, radical life extension and astrobiology.

Protein function prediction is, and will remain, an important research topic with fruitful applications in both the public and private sectors.

For my parents, Siân and Tim,

and my partner, Gal.

Acknowledgements

I would particularly like to thank Christine Orengo, my supervisor, for her guidance, encouragement and understanding throughout my PhD.

I would also like to thank my other supervisors, Jürg Bähler, Rob Finn, Jon Lees, John Shawe-Taylor and Kostas Thalassinos for helping to shape my research projects.

The work in this thesis would not have been possible without the incredible contributions from my collaborators, Nico Bordin, Adam Cryar, Sayoni Das, Fiona Kerr, Clemens Rauer, Maria Rodriguez Lopez and Ian Sillitoe.

I would also like to extend my thanks to all members of the CATH group, past and present, including Mahnaz Abbasian, Tolu Adeyelu, Sebastian Applewhite, Paul 'Ash' Ashford, Joseph Bonello, Sean Le Cornu, Natalie Dawson, Su Datt Lam, Tony Lewis, Millie Pang, Neeladri Sen, Vaishali Waman and Laurel Woodridge.

I am grateful to Rob Finn and Janet Thornton for hosting me at the European Bioinformatics Institute, and to Alex Almeida, Martin Hölzer, Sara Kashaf, Alex Mitchell, Lorna Richardson, Paul Saary and all members of the Metagenome Informatics and Sequence Families teams for making my time there so enjoyable.

My heartfelt appreciation goes to my partner, Gal Horesh, for her loving support, boundless encouragement and brilliant scientific ideas, which kept me going over the past four years.

Special thanks to my friends, Eddie Anderton, Emma Elliston and Archie Wall.

I would like to recognise my undergraduate tutors at Oriel College, Max Crispin, Lynne Cox and Shona Murphy, without whom I would not have embarked on this journey.

Finally, I am grateful to Wellcome for funding me.

Thanks also to my examiners, Mark Wass and Nick Luscombe, for reading my thesis so thoroughly, asking interesting questions and making the viva so enjoyable.

Contents

List of figures

List of tables

List of abbreviations

Abbreviation	Phrase
Aβ	amyloid beta
ABH	α/β hydrolase
AD	Alzheimer's disease
ANN	artificial neural networks
APP	Aβ precursor protein
AUPR	area under the precision-recall curve
Aβ	amyloid beta
Aβ42	42 amino acid long amyloid beta
BIC	Bayesian information criterion
BLAST	basic local alignment search tool
BP	biological process
CAFA	Critical Assessment of Functional Annotation
CART	classification and regression tree
CC	cellular component
CCS	collision cross section
CNN	convolutional neural network
DAG	directed acyclic graph
deepNF	deep network fusion
DIA	data-independent acquisition
DOPS	diversity of position scores
E-value	expect value
EC	Enzyme Commission
ESI	electrospray ionisation

FAD	familial Alzheimer's disease
FDR	false discovery rate
FWER	family-wise error rate
FYPO	Fission Yeast Phenotype Onotology
GO	Gene Ontology
GOLD	Genomes OnLine Database
HMM	hidden Markov model
HPLC	high performance liquid chromatography
I. sakaiensis	*Ideonella sakaiensis*
LC	liquid chromatography
LCC	leaf-branch compost cutinase
LSTM	long short-term memory
MAG	metagenome-assembled genome
MALDI	matrix-assisted laser desorption/ionisation
mAUPR	micro-averaged area under the precision-recall curve
MAUPR	macro-averaged area under the precision-recall curve
MDA	multi-domain architecture
MDAE	multimodal deep autoencoder
MF	molecular function
MHET	mono(2-hydroxyethyl) terephthalic acid
MIPS	Mammalian Protein-Protein Interaction Database
MLP	multi-layer perceptron
MS	mass spectrometry
MS/MS	tandem mass spectrometry
MSA	multiple sequence alignment
NFT	neurofibrillary tangle
ORF	open reading frame
PCA	principal component analysis
PET	polyethylene terephthalate
PMBD	Plastics Microbial Biodegradation Database
PSI-BLAST	position-specific iterative BLAST
PSSM	position-specific scoring matrix

ReLU	rectified linear unit
RF	random forest
RMSD	root-mean-square deviation
RNN	recurrent neural network
S. cerevisiae	*Saccharomyces cerevisiae*
S. pombe	*Schizosaccharomyces pombe*
SAD	sporadic onset Alzheimer's disease
SDP	specificity-determining positions
SGD	stochastic gradient descent
SSAP	structure and sequence alignment program
STRING	Search Tool for the Retrieval of Interacting Genes/Proteins
SVM	support vector machine
SXX	XX% sequence identity
TgAD	transgenic fly line expressing human Arctic mutant Aβ42
UPGMA	unweighted pair group method using arithmetic averages
UPLC	ultra performance liquid chromatography

List of publications

- Sillitoe, I., Dawson, N., Lewis, T. E., Das, S., Lees, J. G., Ashford, P., Tolulope, A., Scholes, H. M., Senatorov, I., Bujan, A., Rodriguez-Conde, F. C., Dowling, B., Thornton, J. & Orengo, C. A. "CATH: expanding the horizons of structure-based functional annotations for genome sequences." *Nucleic Acids Research* (2019)
- Scholes, H. M., Cryar, A., Kerr, F., Sutherland, D., Gethings, L. A., Vissers, J. P. C., Lees, J. G., Orengo, C. A., Partridge, L. & Thalassinos, K. "Dynamic changes in the brain protein interaction network correlates with progression of Aβ42 pathology in *Drosophila*." *Scientific Reports* (2020)
- Lam, S. D., Bordin, N., Waman, V. P., Scholes, H. M., Ashford, P., Sen, N., van Dorp, L., Rauer, C., Dawson, N. L., Pang, C. S. M., Abbasian, M., Sillitoe, I., Edwards, S. J. L., Fraternali, F., Lees, J. G., Santini J. M. & Orengo, C. A. "SARS-CoV-2 spike protein predicted to form stable complexes with host receptor protein orthologues from mammals." *Scientific Reports* (2020)
- Das, S., Scholes, H. M., Sen, N. & Orengo, C. A. "CATH functional families predict functional sites in proteins." *Bioinformatics* (2020)
- Sillitoe, I., Bordin, N., Dawson, N., Waman, V. P., Ashford, P., Scholes, H. M., Pang, C. S. M., Woodridge, L., Sen, N., Abbasian, M., Le Cornu, S., Lam, S. D., Berka, K., Hutařová Varekova, I., Svobodova, R., Lees, J. G. & Orengo, C. A. "CATH: increased structural coverage of functional space." *Nucleic Acids Research* (2021; in press)

A theory is something nobody believes, except the person who made it. An experiment is something everybody believes, except the person who made it.

Albert Einstein

Chapter 1

Introduction

1.1 Protein function prediction

We know the sequences of many proteins, but we do not have a complete picture of the functions that these proteins carry out. Furthermore, whilst the number of known protein sequences grows exponentially larger, the experimental characterisation of functions increases approximately linearly. The knowledge gap is wide and continues to widen. Closing this gap would require an exponential increase in money, materials and manpower—which is unlikely to happen. Instead of closing the gap *experimentally*, functions could be assigned to proteins *predictively*. This is the aim of protein function prediction.

Before we proceed to use the term 'function' throughout this thesis, it seems appropriate to define what the word means. The definition of function is ambiguous and the meaning varies depending on the context in which it is used [1, 2]. Function is often used in the context of proteins, but other biological molecules have functions, for example small RNAs control many cellular processes, DNA is incorporated into biofilms, and asymmetric distributions of lipids in the cell bilayer signal the health of cells. Proteins, however, seem to be the most versatile biopolymer with respect to the range of functions that are possible. Furthermore, protein function can be augmented by post-translational modifications, such as lipids, sugars and phosphate groups, to generate subtle variations in function and thus increase an organism's functional reservoir.

The concept of function extends over many different levels [1, 3]. At the microscopic level, proteins have molecular functions, such as catalytic activities, binding sites and allosteric sites. Cellular biological processes constitute function at the mesoscopic level, including metabolic and signalling pathways. Finally, function manifests at the macroscopic level of organisms. (Arguably, the macroscopic level also includes ultra-individual func-

tions, such as the behaviour of multiple organisms, from the same, or different, species.) Lower-level biochemical functions can contribute to physiological functions on higher levels [2]. Disruption of lower-level functions, such as through mutation, can cause catastrophic organismal phenotypes, such as in the case of cancer. Therefore, an appropriate vocabulary must be used when referring to function.

1.2 The pre-bioinformatic age

In the pre-bioinformatic age—before genomics, structural biology and high-throughput function assays became routine—characterising the functions of proteins was a laborious task. To do so required painstaking molecular biology experiments, involving random mutagenesis by radiation, site-directed mutagenesis by PCR or observation of naturally occurring variants, combined with assays to test for specific functions. These were slow, expensive and low-throughput. It is almost impossible now to imagine performing molecular biology experiments before recombinant protein expression was possible. But this was the case until 1976 [4].

Nature has evolved a multitude of proteins that participate in myriad molecular processes required for life. Some processes are always required, whereas, others are regulated by environmental conditions, chemical signals or stress responses. There is a tendency to think of proteins as microscopic machines that fulfil the requirements for life, however, this is simply wrong. DNA, RNA, lipid and polysaccharide biopolymers are also vital to life, functioning in concert with proteins. Furthermore, proteins are not static tangles of amino acids, but rather are in constant flux, due to alternate splicing, post-translational modifications and allosteric changes. Nature is parsimonious, so the dynamic nature of these molecules must exist for a reason. By creating a cellular milieu, the effective number of molecules encoded in the genome is increased, the functions of these molecules can be fine-tuned, and their activity can be regulated precisely.

Phrasing this in the language of protein function prediction: given a protein P and the set of all known biological functions F, the task is to assign one or more functions $\{F_1, ..., F_k\} \subseteq F$. Solutions to this problem are:

- **Sparse**. Nature has evolved according to the principles of 'division of labour'—many proteins are required for life—and 'task specialisation'—each protein only carries out a subset of all possible functions.
- **Redundant**. Relationships between functions can be described using a taxonomy,

which allows redundant information to be omitted. Knowledge of one function may imply one or more other functions. For example, a peptidase enzyme, that hydrolytically cleaves peptide bonds, is both a hydrolase and a peptidase, where peptidase activity is a more specific functional description than hydrolase activity.

- **Disjoint**. A protein may have multiple, unrelated functions. Proteins are often composed of multiple domains, whose arrangement is linear with respect to the primary protein structure, but may fold differently in space. The 3D conformation may also change in time, thus giving rise to a 4D problem. Whilst each domain typically has tightly defined functions, combinations of, and interactions between, domains can give rise to new functions [5, 6].
- **Poorly defined**. Protein function prediction can only predict functions that have already been identified. Many more functions may exist in nature than are currently known. Factors contributing to this phenomenon include: experimentally validating functions is time-consuming; vast numbers of proteins have been sequenced; biases in which sequences, organisms and environments have been studied; inability to culture organisms, particularly microbes identified in metagenomics; and neglect of studying extreme or non-natural environments, that have high selection pressures for the evolution of new functions.
- **Noisy**. Biological data is inherently noisy, due to the stochastic nature of physical phenomena.

The ultimate aim of protein function prediction is to maximise the number of true functions predicted, to predict these functions as specifically as possible, and to minimise the number of incorrect functions predicted. Previously, there was no programmatic way to predict protein function. However, the burgeoning field of bioinformatics began to develop computer science techniques, statistical methods, and data-driven approaches to tackle the problem of protein function prediction—and to do so on a large scale.

1.3 The dawn of bioinformatics

Bioinformatics is a wonderfully non-specific name, that imparts minimal knowledge about what the eponymous field does, or how it does it. One could argue that this diffuse definition has been the secret of its success, as it has allowed ideas to be integrated from a diverse range of fields, including—but not limited to—computer science, mathematics, statistics, machine learning, graph theory, data science, linguistics, computational simula-

tions, mechanics and image processing. Within bioinformatics, protein function prediction has predominantly relied on statistics, machine learning and graph theory to detect signals and patterns in biological data.

No bioinformatician is an island entire of itself. Bioinformatics depends intimately on other types of science and scientists. First, data must be collected by experimentalists—there is currently no substitute for this. Although, in the future it is conceivable that, given sufficient understanding of biology and enough compute power, entire experiments could be simulated *in silico*. Second, data is converted to knowledge by talented database curators who store, link and annotate experimental data. Without these other disciplines, bioinformatics would not be as useful as it is today.

The holy grail of protein function prediction is to be able to predict functions *ab initio*. To do so will require an almost complete molecular description of life, which we appear to be a long way away from. For now, protein functions are predicted transitively. For example, given two proteins P_1 and P_2 that are sufficiently similar, let F be a function that has been experimentally assigned to P_1. By the transitive property, F can also be assigned to P_2. This process is referred to using different names, including, but not limited to, 'homology transfer' and 'guilt by association'.

In biology, homology refers to similar features that have a common ancestry and are inherited through evolution. A pair of proteins are said to be homologous if they share a common ancestor. In such cases, transferring F from P_1 to P_2 may be appropriate if P_1 and P_2 are homologous. Generally speaking, $> 60\%$ sequence identity—the proportion of identical residues between a pair of proteins—is required for conservation of function, as measured by being able to transfer Enzyme Commission (EC) functional annotations entirely with $\geq 90\%$ accuracy [7, 8].

However, it is not always safe to transfer functions in such ways. Homologous proteins can be subdivided into orthologous sequences—referring to evolutionary specialisation of sequences—and paralogous sequences—meaning duplication of sequences within an individual, followed by divergence of function. Therefore, similar sequences do not necessarily have similar structures or functions [1, 9], so care must be taken when transferring functions between homologous proteins [10, 11].

The 'orthologue conjecture' posits that "orthologous genes share greater functional similarity than do paralogous genes" [12], so inheritance of function is safer between or-

thologues than paralogues [13]. This claim became largely accepted within the biological community, without substantive evidence to back it up. Subsequently, various studies have attempted to test the conjecture. One study found that, after controlling for multiple biases in function annotation data sets, there is a weak effect that orthologues have more similar functions than paralogues [14]. Another study found that orthologues and paralogues evolve and diverge at similar rates, so orthologues may be as likely to have different functions as paralogues [15]. Recent work shows that there are actually two orthologue conjectures: one being a statement about the evolution of function—which is difficult to test—whilst the other is a statement about the prediction of function [16]. The authors found that function prediction is improved when the amount of data is maximised by using both orthologue and paralogue data [16].

Furthermore, biological data is noisy and incomplete. F may not actually be a function of P_1 (a false positive). If F is not a function of P_1, it might, or might not, be a function of P_2. If care is not taken, errors can be easily propagated through biological databases. These errors are notoriously difficult to correct.

To allay fears about the rather large and powerful elephant in the room, we focus on machine learning methods in Section 1.4 and on the application of these methods to protein function prediction in Section 1.5. For now, we focus on classical methods to annotate proteins with functions. Overwhelmingly, the most popular method for protein function prediction is by sequence and structure homology, which we explain next.

1.3.1 Sequence homology

Sequence homology methods for protein function prediction rely on identifying related proteins whose functions have already been characterised [1, 3, 11, 13]. Typically, sequence alignment and sequence clustering methods are exploited to cluster proteins into evolutionarily related groups.

Basic local alignment search tool (BLAST) [17] is an incredibly popular method to search a sequence database with a query sequence to find similar sequences. BLAST first divides the query sequence up into short k-mers, or words. Each target sequence in the database is searched for high-scoring matches to these words, using the BLOSUM62 substitution matrix. Exact matches are extended to form high-scoring segment pairs between the query and target sequences, which are retained only if their score is high enough. Two or more high-scoring segment pairs are then joined to make longer alignments using the

Smith-Waterman local sequence alignment algorithm. Target sequence hits are returned by BLAST, and any annotations that they contain can be transferred to the query sequence to predict its function. BLAST became, and remains, popular due to its advanced statistical framework, based on the expect value (E-value). Given a database of a particular size and a query sequence, the E-value of a hit measures the number of sequences that one can expect to see in the database by chance with the same match score. The lower the E-value, the more significant a match is. As such, the E-value can be used as a threshold to filter out the random background noise when searching a database.

As BLAST is a heuristic method, it is advantageous for searching large sequence databases. However it can miss harder to find matches, such as those found in distantly related homologues. A position-specific scoring matrix (PSSM), also known as a profile, can be generated from multiple sequence alignments (MSAs) to represent the statisical properties of sequences and motifs. PSSMs are matrices generated from MSAs. Given an alignment of length n, constructed from an alphabet of symbols Σ, the corresponding PSSM is an $[|\Sigma| \times n]$ matrix with normalised frequencies of each symbol at each position. The columns in this matrix describe the propensity for the symbols at each position in the sequence, and therefore the substitution probabilities. Position-specific iterative BLAST (PSI-BLAST) [18] uses profiles to find more distantly related sequences than BLAST can find [19]. PSI-BLAST first builds a profile from closely related sequences, then searches the sequence database using the profile. This process can be run iteratively, building profiles out of the hits from the previous iteration to find more distantly related sequences.

Proteins are composed of one or more domains, often arranged as a linear sequence in the primary structure. Domains are protein structure units that are stable and can fold independently of the wider protein context. Combinations of domains can give rise to novel functions [5, 6]. The linear arrangement of domains in protein sequences—known as the multi-domain architecture (MDA)—and the organisation of these domains in 3D space, are determinants of function [20–23]. Sequence-based domain resources, such as Pfam [24] organise domains into homologous domain families. In the larger Pfam families, functions of members can be highly divergent between paralogues and distant orthologues [11].

1.3.2 Hidden Markov models

Functions can be transferred within a protein domain family, from one member whose function has been characterised, to all other members of the family. MSAs can be used to infer the phylogeny that relates domains into domain families, through evolutionary time. The sequence diversity in an MSA can be represented using a hidden Markov model (HMM) [25, 26]. HMMs are statistical models of Markov processes, defined as processes where the next state of a system depends only on its current state.

Given an MSA, the transitions between states in the HMM are trained using the observable distribution of amino acid, gap and deletion probabilities in the MSA (Fig. 1.1). A sequence can be scanned against a family's HMM to score whether the sequence matches the family, and therefore whether the sequence is evolutionarily related to the sequences in the family.

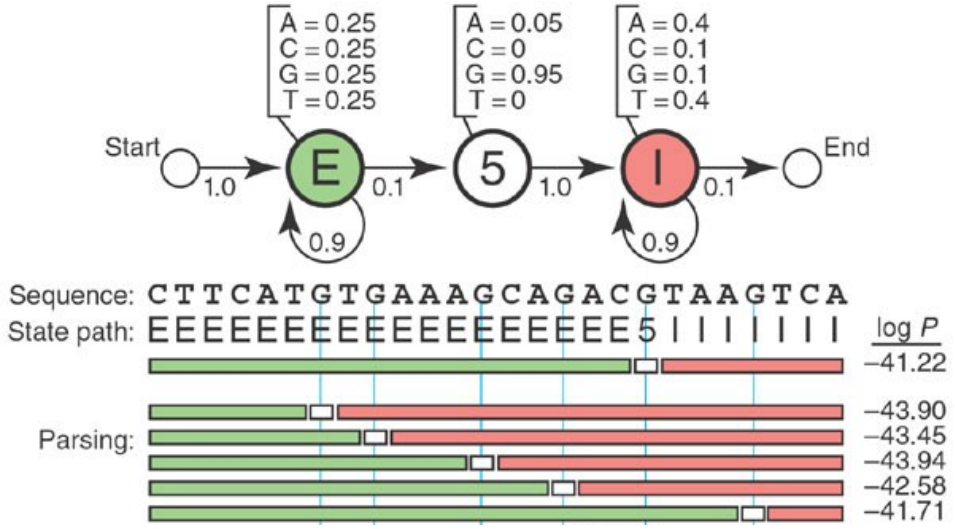


Figure 1.1: **Anatomy of a hidden Markov model (HMM) that models the sequence dependencies of a 5' splice site in DNA.** States for the exon (E), 5' splice site and intron (I) are shown. Transition probabilities determine the paths that are allowed between the states, with one or more nucleotide in the exon, followed by the 5' splice site at a G or A nucleotide, followed by one or more nucleotides in the intron. Emission probabilities are shown above the states that model the nucleotide composition of sequences in the three states. Nucleotides are emitted at every state visited on a path through the model from Start to End. Potential paths through the model for the 'Sequence' are shown in the 'Parsing' section, where the 5' splice site state is a G or A. Log probabilities of paths are calculated by multiplying all transition and emission probabilities together and taking the logarithm. The most likely path is the path with the highest probability, in this case $\log P = -41.22$, which corresponds to the true 'State path' for the sequence. Figure adapted from [26].

HMMER [27] and HHsuite [28] are the de facto standard tools to build HMMs and scan

sequences against them. HHsuite can also perform pairwise HMM alignment [29], which can be thought of as comparing two MSAs. The jackhmmer program in HMMER, akin to PSI-BLAST, is an iterative search tool. Given a single query sequence and a target sequence database, an HMM is built from the sequence using a position-independent substitution matrix, such as BLOSUM62. Sequences in the target database are scanned against the HMM to identify hits, which are then aligned and used to build a new HMM. The whole process can be repeated for a few iterations to pull in ever more distantly related sequences.

1.3.3 Structure homology

We saw in the previous sections on sequence homology and HMMs that the sequence of amino acids—the primary protein structure—can be used to predict function. Therefore, sequence-function relationships exist. In this section, we extend this concept to structure-function and sequence-structure-function relationships [3, 11, 13].

The primary protein structure can fold, which allows proteins to adopt higher order structures with many degrees of freedom. A 100 residue protein, with 99 peptide bonds, has approximately $3^{198} = 3 \times 10^{95}$ stable ϕ and ψ bond angle conformations [30]. The polypeptide chain folds into local secondary structures [31], which arrange in space relative to each other in the tertiary structure. Multiple polypeptide chains can interact with each other in space to form the quaternary structure.

Higher order protein structures can be used to predict protein function [3, 13]. The protein structure imparts knowledge about the 3D arrangement of amino acids to form functional sites in proteins, such as catalytic sites, regulatory motifs and allosteric sites [2].

Structure is more conserved than sequence [32], which is perhaps to be expected, given the evolutionary need for proteins to be able to form stable arrangements in three-dimensional space. Distantly related proteins, whose sequences have diverged considerably, so that they share lower sequence similarity than can be reliably detected using sequence comparison methods, may still be identified as homologues by comparison of their structures [11]. Therefore, structural homology can be used to predict protein function. Conservation of structure between two proteins can be identified by measuring the root-mean-square deviation (RMSD) between equivalent atomic positions in two protein structures aligned in three-dimensional space.

CATH [33] (introduced in Section 1.3.4) and SCOP [34] classify domains into

structurally-related homologous superfamilies, akin to Pfam for sequences. Both resources organise structures in a hierarchical taxonomy, that range from describing abstract structural features near the root, to specific structural features of superfamilies. Superfamilies are sub-classified into families in SCOP and functional families, or FunFams, in CATH (introduced in Section 1.3.4.2). Sequences in FunFams may not share high sequence identity, but critical residues—so called specificity-determining positions (SDPs)—will be conserved. This is not true for SCOP families, which are formed purely by sequence identity clustering. As such, SCOP families are more similar to 'S60 clusters' (60% sequence identity clusters) of CATH superfamily domain sequences.

Knowledge of a protein's structure is neither necessary or sufficient to predict its function [13, 35]. Historically, a protein's structure would be solved only after its function had been experimentally characterised. Like with sequence homology, care must be taken with transferring functions between structural homologues. Structural homology is sometimes sufficient for proteins to share function. However, some structures that have similar structures perform different functions. For example, the TIM barrel fold has a large pocket that has been exploited in multiple ways by introducing mutations in lining of the pocket that augment the function of the protein [36, 37]. Over 27 CATH superfamilies [33] contain a TIM barrel fold, covering > 60 different EC terms [11]. Conversely, structural homology is not necessary for proteins to share function. Some proteins, like the many unrelated families of serine proteases, have separately evolved the catalytic triad via convergent evolution.

1.3.4 CATH

CATH [33, 38, 39] classifies protein domain structures into evolutionarily related families, arranged in a four-level hierarchical taxonomy:

1. **C**lass: secondary structure, all alpha-helical, all beta-sheet, mixed alpha-helical and beta-sheet, or little secondary structure.
2. **A**rchitecture: global arrangement of secondary structure elements into the tertiary structure.
3. **T**opology/fold: specific arrangement of secondary structure elements.
4. **H**omologous superfamily: evidence of evolutionary relatedness of domains.

The current version of CATH (v4.2) contains 6,119 superfamilies. To identify these groups, protein structures are aligned using SSAP (structure and sequence alignment pro-

gram) [40]. SSAP employs double-dynamic programming to extend the concept behind Needleman and Wunsch's sequence alignment algorithm [41] to 3D protein structures. As such, the double-dynamic programming method is guaranteed to find the optimal alignment of any two given proteins. The algorithm was subsequently modified to align structural motifs [42], akin to Smith and Waterman's adaptation of the global sequence alignment algorithm to find local matches [43].

Crucial to its success is the way that SSAP represents protein structures. Instead of using the letters of the amino acid alphabet, SSAP uses interatomic vectors between the Cβ atoms of all residues, except glycines. The inclusion of positional information proved key to SSAP's success over contemporary methods that only considered interatomic distances [44]. Furthermore, interatomic vectors proved to be necessary and sufficient to align protein structures [45]. Inclusion of additional structural information improved alignments in only the most challenging cases [45].

In 1993, there were structures of $1{,}800$ chains in the PDB. Because ~ 50 new structures were being deposited each month, it became obvious that an automated method would be needed to identify protein folds and classify proteins into protein fold families. SSAP was initially used to align proteins containing globin domains that have very low sequence similarity, but were found to conserve the same domain structure [40]. SSAP was subsequently used to identify families of protein folds in groups of proteins with similar sequences [46]. These families established the foundations of CATH [38, 39].

CATH can be used for protein function prediction in a number of ways. Firstly, CATHEDRAL [47] can be used to align a query structure to structure representatives from each superfamily. Functions of proteins within any matching superfamilies can be transferred. However, this classification is usually too broad to provide fine-grained functions. Secondly, CATH-Gene3D can be used to assign a protein sequence to CATH superfamilies (see Section 1.3.4.1). Finally, CATH-FunFams can be used to assign a protein sequence to FunFams (see Section 1.3.4.2). Next, we introduce these two methods Gene3D and FunFams.

1.3.4.1 Gene3D

Gene3D [21, 48, 49] allows CATH superfamily domains to be predicted in protein sequences, using sequence information alone. Gene3D exploits protein sequence-structure relationships by representing the sequence diversity of each CATH superfamily as a

set of one or more HMMs. Sequences are scanned against these HMMs and assigned to any matching superfamilies. Interestingly, Gene3D is a hybrid sequence-structure-homology method because structures are used to define the domain boundaries of domain sequences. The equivalent functionality of Gene3D is provided to SCOP via the SUPERFAMILY database [50].

To construct Gene3D's HMMs, domain sequences within a CATH superfamily are first clustered at S95 using CD-HIT [51], aligned and an HMM is built using jackhmmer's iterative search functionality [27]. UniProt is searched using this HMM and any sequences that meet the HMM inclusion threshold are retained. A second iteration of jackhmmer is then performed. The HMM from the second iteration is used if the ratio of true positive predictions improves, otherwise the HMM from the first iteration is used.

Gene3D supports the identification of discontinuous domains. Discontinuous domains can form as the result of insertion events, whereby a continuous domain becomes discontinuous when a domain is inserted between its start and stop positions. To improve their identification, HMMs are built for discontinuous domains by also including the inserted domain sequences.

In any given protein sequence, there may be many, overlapping, matches to S95 models from the same, or different CATH superfamilies. cath-resolve-hits [52] is used to reduce the set of matches to an optimal subset of non-overlapping matches (Fig. 1.2). The algorithm uses dynamic programming to deterministically find the set of domains that maximises the sum of the bit scores.

1.3.4.2 FunFams

FunFams are groups of homologous proteins that perform the same function, or functions. SDPs are used to identify and group likely isofunctional proteins into FunFams. Sequences in CATH superfamilies are sub-classified into fine-grained groups, according to sequence alone. The current version of CATH (v4.2) contains 67,598 FunFams in 2,620 of the 6,119 superfamilies. Whilst FunFams are strictly a type of sequence-homology method, the sequences used to construct CATH-FunFams are from Gene3D hits, so are structural homologues. FunFams were only made possible because Gene3D is now able to retreive millions of sequence homologues from UniProt. Having said this, FunFam is a general sequence-based protocol that can be applied to any protein family. In addition to CATH-FunFams, we also generate Pfam-FunFams from Pfam families.

Figure 1.2: **cath-resolve-hits resolves the domain boundaries of multiple 'Input Hits' to an optimal subset of non-overlapping 'Resolved Hits' that form the MDA.** Figure taken from https://cath-tools.readthedocs.io/en/latest/tools/cath-resolve-hits/.

FunFams are generated in a four step process:

1. **Sequences containing Gene3D predictions of a particular superfamily domain are separated by MDA and clustered at S90**

   Clusters that do not have any experimental GO terms associated are removed. The remaining clusters are referred to as starting clusters. Clusters that are not associated with any experimentally characterised functions are removed for practical—not scientific—reasons. Removing these functionless clusters helps to reduce the total number of starting clusters, to fit within memory limits. Given unlimited memory, or a low-memory method of generating FunFams, FunFams could be generated using all S90 clusters. This would mean that some FunFams would not have any associated functions, but these could always be added *post hoc* once any one of the FunFam's members became experimentally characterised.

2. **GeMMA [53] is a greedy algorithm that builds a neighbour-joining tree by bottom-up hierarchical agglomeration**

   Given n starting clusters, GeMMA makes $n-1$ merges. Leaves and internal nodes in the tree are clusters of related superfamily domain sequences, represented as HMMs (Fig. 1.3). To decide which node to merge at each of the $n-1$ steps, HMMs are aligned pairwise and the closest pair of clusters are merged. The merged set of sequences are aligned, a new HMM is generated and the merging process is repeated.

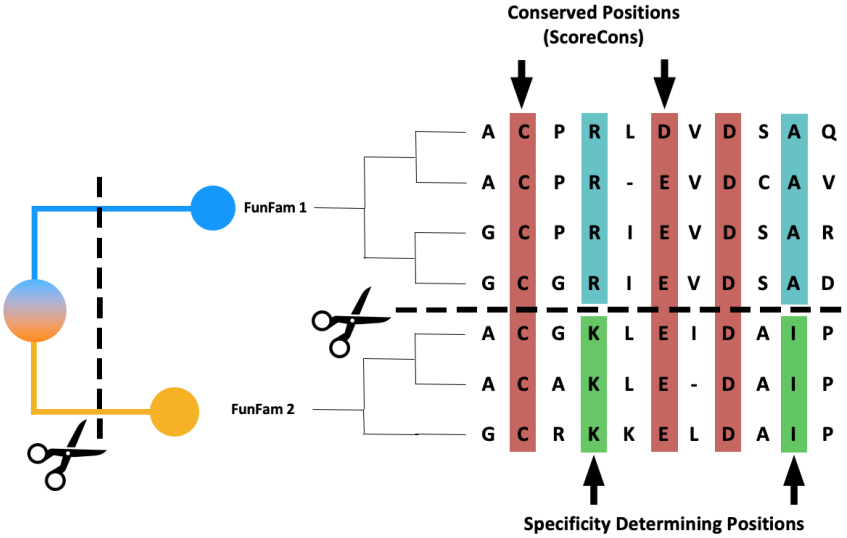


Figure 1.3: **An overview of the GeMMA and FunFHMMer algorithms.** Scissors denote the points where the FunFHMMer algorithm cuts the GeMMA tree. Specificity-determining positions are identified using GroupSim (Fig. 1.4). Figure courtesy of Nicola Bordin.

Generally, we have noticed that the upper bound of starting clusters is $5{,}000$ before reaching the memory limits of our largest machines (with 3 TB memory). GeMMA used to be applied to an entire superfamily [53]. As superfamilies have grown in size, the protocol was changed so that GeMMA is run on subsets of proteins that have the same MDA (see Section 1.3.4.3). Partitioning superfamilies by MDA makes biological sense because the MDA is a determinant of function [20–22]. Proteins with different MDAs are unlikely to be in the same FunFam, so it is reasonable to segregate them *ab initio*.

3. **FunFHMMer [54] determines the optimal partitioning of the GeMMA tree into clades, each of which is a FunFam**

   FunFHMMer operates on MSAs of leaves and internal nodes (Fig. 1.3). Diverse protein sequences are required for FunFHMMer to elucidate conservation patterns and SDPs in MSAs. FunFHMMer traverses the GeMMA tree from the leaves towards the root. Let v_i and v_j be two child nodes that are connected to a parent node v_p. A functional coherence index is calculated for v_p to determine whether the tree should be cut before v_p (traversing in the direction from leaf to root). If the tree is cut before v_p, two FunFams F_1 and F_2 are produced, where $F_1 = v_i$ and $F_2 = v_j$. Otherwise, if the tree is subsequently cut after v_p, v_i and v_j will form part of the same FunFam F, $\{v_i \cup v_j\} \subseteq F$. The functional coherence index is powerful, generates func-

tionally pure FunFams and imbues FunFams with their predictive power. The index considers three parameters:

- Information content of the MSA. Calculated using the diversity of position scores (DOPS) from ScoreCons [55]. MSAs with DOPS > 70 are generally considered to be sufficiently diverse.
- Proportion of predicted SDPs in an MSA. SDPs are predicted using GroupSim (Fig. 1.4), which calculates a score for each position in an MSA, whose sequences are pre-assigned into two groups according to the sequences in the child nodes [56]. The SDP-to-conserved position ratio determines whether the tree is cut.
- Gaps in an MSA. Gaps in the parent alignment indicate that the child alignments were of different lengths. A multiplicative factor of 0 results if the number of gap positions is greater than the number of non-gap positions.

Some residues that are necessary for the function of a protein, such as catalytic residues in the active site of an enzyme, may be highly conserved in a set of S90 clusters. Whilst these residues may be predictive of the general function of a given protein, they do not determine how the GeMMA tree is partitioned by FunFHMMer into FunFams. Only differentially conserved residues determine the partitioning of the tree, and therefore the functional specificity of FunFams. Highly-conserved residues may also be useful for protein structure prediction, particularly for evolutionary covariation techniques [57, 58].

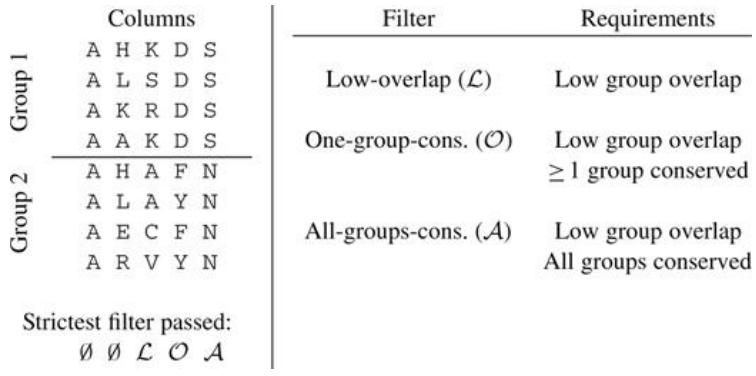


Figure 1.4: **GroupSim predicts specificity-determining positions (SDPs) in alignments.** Each column in the alignment is assigned to one of four sets, explained on the right of the figure, where $\emptyset$ denotes the empty set. Figure taken from [56].

4. **Generate full FunFams**

The FunFams generated by the previous step are known as the seed alignments. Seed alignment sequences are then scanned against their corresponding HMM to assign an inclusion threshold to each FunFam, which is the largest E-value for the worst match. Full alignments for FunFams are generated using sequences from the superfamily that were in S90 clusters without an experimental GO term. These sequences are scanned against the seed HMMs using the per FunFam inclusion thresholds. Sequences are assigned to any matching FunFam, but if sequences match to multiple FunFams, they are assigned to the best matching FunFam.

1.3.4.3 GARDENER

The FunFam generation protocol has recently been improved in a new algorithm, GARDENER, which performs two rounds of FunFamming, rather than the single round in the original protocol set out above (Fig. 1.5). In the first round, GARDENER processes each MDA partition separately using GeMMA and FunFHMMer to generate FunFams. These initial FunFams, from all of the MDA partitions, are then pooled. Initial FunFams are then treated as starting clusters for a second round of GeMMA and FunFHMMer. The FunFams from the second round of FunFamming are used as FunFams for the superfamily. Note that if a superfamily has a single MDA, then only one round of GeMMA and FunFHMMer are performed, instead of two. GARDENER is advantageous because it allows over-split FunFams and singleton FunFams, from the first round, to be merged in the second round. GARDENER is explained in more detail in Section 3.2.10.

1.3.5 Symbolic representations of functions

So far, we have been referring to functions as abstract concepts (Section 1.1). Here, we introduce data structures that are used to represent functions symbolically. In recent years, natural language processing of unstructured text has become increasingly powerful, but for computational and curational ease, functions are best represented in a structured way. These data structures must represent the set of all possible functions (or all known functions), rather than assigning functions *ad hoc* [11]. One way to do this is using a 'controlled vocabulary' for the set of all possible functions.

We have already noted that functions are related to each other (Section 1.2). Some functions are more closely related to some than to others. As such, controlled vocabularies of functions lend themselves naturally to being organised in hierarchical, tree-like structures. The Enzyme Commission (EC) [59] uses a simple four-level hierarchy, where

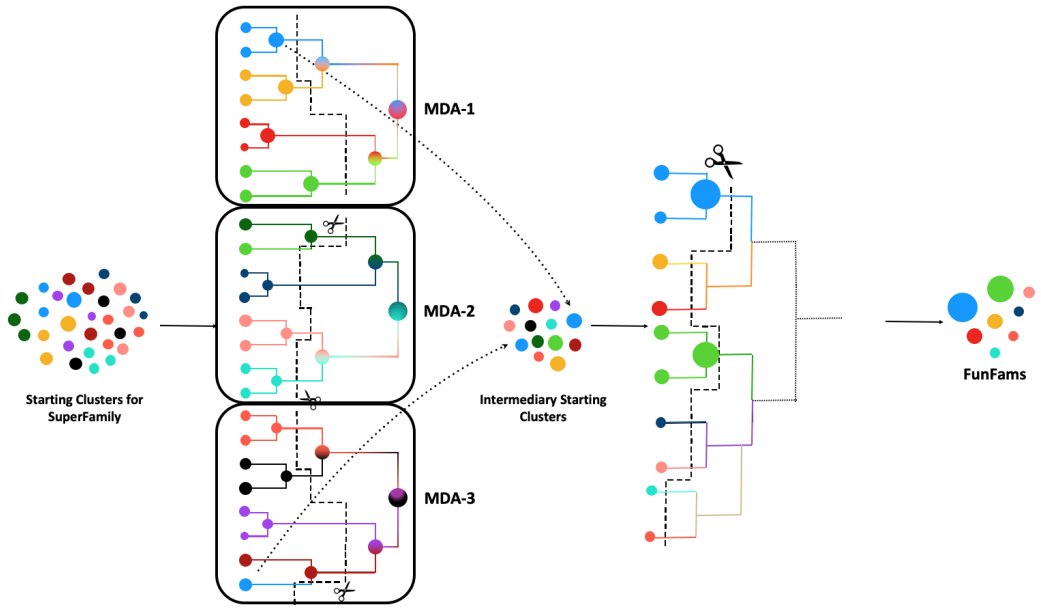


Figure 1.5: **An overview of the GARDENER algorithm.** Scissors denote the points where the FunFHMMer algorithm cuts the GeMMA tree. Figure courtesy of Nicola Bordin.

child terms are strict descendants of one parent term. Child terms describe more specific functions that are a subset of the parent function. For example, the function of tripeptide aminopeptidases are described by the EC number "EC 3.4.11.4", where the four levels correspond to:

- EC 3 hydrolases
- EC 3.4 peptidases
- EC 3.4.11 N-terminal exopeptidases
- EC 3.4.11.4 N-terminal exopeptidase for tripeptides

7,936 different enzymatic functions have been described using the EC database. These terms have been annotated 246,858 times to a total of 235,086 protein IDs from direct experimental characterisation of an enzyme's activity, reactants, products, cofactors and specificities.

Whilst EC terms are focussed on enzyme function, the Gene Ontology (GO) GO [60] is a more ambitious project than EC, in that it aims to characterise molecular function, biological processes and subcellular location of proteins. So far, the GO describes 44,167 functions, across three disjoint namespaces: 'biological process' (BP), 'molecular function' (MF) and 'cellular component' (CC). Proteins from 4,728 species have been characterised using a total of 8,047,744 GO annotations to 1,569,827 unique proteins. We use the GO extensively in this thesis and attempt to increase the functional coverage of proteins by

predicting GO term annotations.

'Ontology' is the branch of philosophy that deals with the nature of being, of which there are many possible relationships. Describing these relationships requires a more flexible data structure than the tree used in EC. The GO uses a directed acyclic graph (DAG) (Fig. 1.6), where edges in the graph are directed from child terms to parent terms. The DAG is actually a multi-DAG (a DAG with multiple edge types), used to represent one-to-many relationships between functions. Possible relationships in the GO DAG include 'is a', 'part of' and 'regulates'. For example, '6-phosphofructokinase activity' is part of 'glycolytic process through fructose-6-phosphate', which can be positively and negatively regulated. Multiple levels of functions can be queried using the GO, for example, it can be used to find all genes in a genome that are involved in signal transduction, or one can focus only on the subset of genes that are tyrosine kinases.

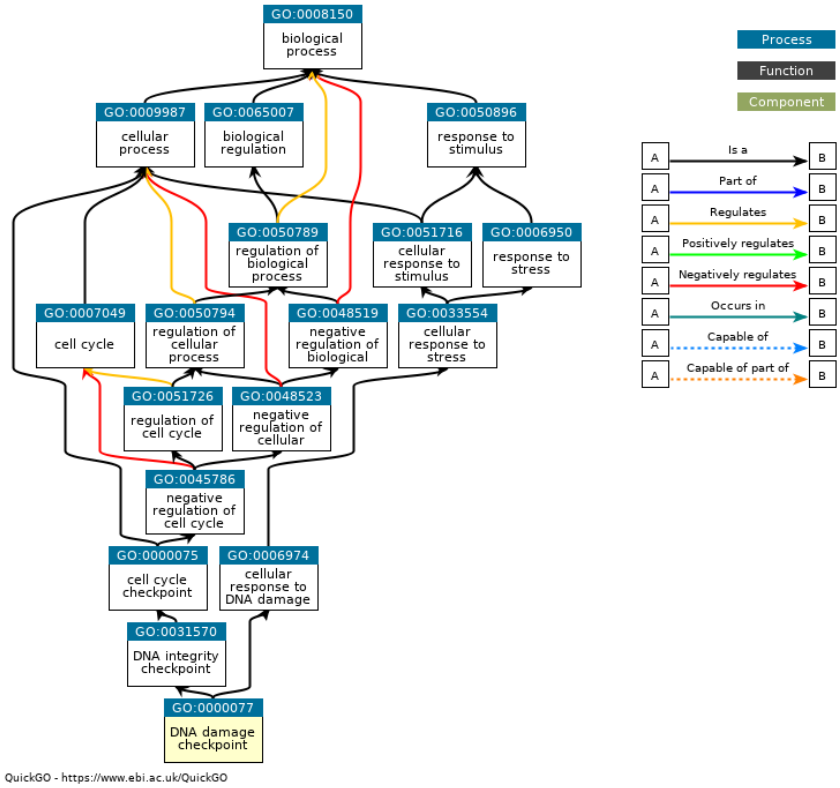


Figure 1.6: **The Gene Ontology.** Subgraph of the Gene Ontology for 'DNA damage checkpoint' (GO:0000077). Figure taken from https://www.ebi.ac.uk/QuickGO/term/GO:0000077.

Whereas EC annotations are based on direct experimental characterisation, and therefore should be trustworthy, GO introduces a notion of quality via its evidence codes attached to each annotation of a GO term to a protein. Evidence codes can be broadly cat-

egorised into groups, with functions assigned by: direct experiment (EXP, IDA, IPI, IMP, IGI and IEP) and their high-throughput counterparts (HTP, HDA, HMP, HGI and HEP); inferred phylogenetically (IBA, IBD, IKR and IRD) or computationally (ISS, ISO, ISA, ISM, IGC and RCA); assigned by a curator (IC); automatically inferred annotations (IEA); and low-quality, untrusted annotations without evidence to back them up (NAS, ND). These groups are approximately ordered by how much confidence one would assign to an annotation with that evidence code.

A number of other protein function databases are touched on in this thesis, including the MIPS FunCat database [61, 62]; the Human Phenotype Ontology [63], used to annotate proteins that cause phenotypic abnormalities in human diseases; and the Disorder Ontology [64] that characterises intrinsically disordered proteins. FunCat was developed during the initial sequencing of the *Saccharomyces cerevisiae* genome to describe yeast protein function. Similar to the EC, MIPS terms are a controlled vocabulary arranged in a three-layer hierarchy of increasing functional specificity. Despite MIPS being rather an ancient resource, we use some of its annotations in Chapter 4 to benchmark our method against two other methods that chose to use MIPS [65, 66]. Although we do not use the Human Phenotype Ontology and the Disorder Ontology, we encounter them in Chapter 5 as part of the CAFA 4 evaluation of protein function prediction methods.

Now that we know how functions are represented computationally, next we introduce how to compare the predictive performance of different methods on the same prediction task.

1.3.6 Benchmarking performance

Many methods have been developed to predict protein function, each often claiming to be the state-of-the-art. Such claims should always be treated with suspicion due to disingenuous science, overfitting on evaluation data, or luck. Transparent, public benchmarks are required to evaluate different methods within a community. Protein structure prediction first introduced the CASP [67] challenge in 1994 to compare structure prediction methods on newly solved structures that were witheld from the community. Structure prediction methods are also continuously evaluated by CAMEO [68]. The CAPRI challenge benchmarks protein-protein docking for structure prediction [69, 70]. DREAM challenges (http://dreamchallenges.org) are a set of various community evaluations of common bioinformatics, genomics and statistics methods, such as for network inference from expression

data [71].

The Critical Assessment of Function Annotation (CAFA) [72–74] is a community evaluation of function prediction methods, metamethods and research groups. Given a set of protein sequences, the task is to predict the most, specific and correct functions for each protein. No restrictions are placed on how this task should be achieved. Participants compete by submitting predictions using any of three disjoint ontologies: Gene Ontology, Human Phenotype Ontology [63], and Disorder Ontology [64].

Methods are evaluated blind, after a sufficient time delay, in which new, experimentally-characterised functions are collected. Proteins are experimentally characterised in an uncoordinated way by a decentralised network of scientists. To all intents and purposes, it is impossible for any participant to cheat by colluding with experimental groups that may have characterised new functions of a set of proteins. Therefore, it can be assumed that no method has been trained using any of the annotations contained in the evaluation set. As such, for a method to perform well on the evaluation set, it must have learnt general patterns that link protein sequence to function from the training data.

Each team can enter predictions from three separate models, which are assessed for coverage and (semantic) precision using $F_{\max}$ and $S_{\min}$. $F_{\max}$ is defined as

$$F_{\max} = \max_{\tau} \left\{ \frac{2 \cdot \mathrm{pr}(\tau) \cdot \mathrm{rc}(\tau)}{\mathrm{pr}(\tau) + \mathrm{rc}(\tau)} \right\}, \tag{1.1}$$

where precision (pr) and recall (rc) are

$$\mathrm{pr}(\tau) = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}$$
$$\mathrm{rc}(\tau) = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}},$$

for true and false positive and negative predictions. $S_{\min}$ is defined as

$$S_{\min} = \min_{\tau} \left\{ \sqrt{\mathrm{ru}(\tau)^2 + \mathrm{mi}(\tau)^2} \right\}, \tag{1.2}$$

where ru is the remaining uncertainty and mi is the misinformation

$$\mathrm{ru}(\tau) = \frac{1}{n_e} \sum_{i=1}^{n_e} \sum_{f} \mathrm{ic}(f) \cdot \mathbb{1}\left(f \notin P_i(\tau) \wedge f \in T_i\right)$$
$$\mathrm{mi}(\tau) = \frac{1}{n_e} \sum_{i=1}^{n_e} \sum_{f} \mathrm{ic}(f) \cdot \mathbb{1}\left(f \in P_i(\tau) \wedge f \notin T_i\right),$$

where $\mathrm{ic}(f)$ is the information content of term f [73].

Teams are ranked according to their best performing model on each metric, which allows teams to enter a 'strict' model that is optimised for $S_{\min}$ and a 'relaxed' model for $F_{\max}$. This highlights another problem related to evaluating function prediction methods: What is the model optimising for? It is hard to declare a clear 'winner' in CAFA because some methods are good according to one metric, but perform poorly according to others. Hamp et al. recognise that

> "For future CAFA experiments, it will therefore become even more important to avoid 'crowning winners' (unless methods stand out by all means) and to focus on method groups suited best for certain disciplines" [75].

1.4 Machine learning

This thesis documents the implementation of multiple models for protein function prediction. The majority of these models were based on machine learning, using an appropriate machine learning algorithm for the task. Supervised machine learning models learnt to predict protein function by identifying general patterns in the training data that are predictive of particular functions. In supervised learning, model performance was optimised by minimising a loss function between the predictions and the empirical target data. Different training and target data were used to optimise models to learn patterns that map protein features in the training data to protein functions in the target data. The machine learning methods used in this thesis were:

- Artificial neural networks, used in Chapter 4
- Support vector machines, used in Chapter 4
- Decision trees, specifically random forests, used in Chapter 5

In this section, we introduce these three machine learning methods, relevant hyperparameters and other considerations.

1.4.1 Artificial neural networks

Artificial neural networks (ANNs) are computational systems that are inspired by the organisation of neurons in biological brains [76]. Neurons receive an input, which gets combined with the internal state of the neuron to produce an output. Optionally, an activation function can be applied to modulate the output in a non-linear way. Neurons are arranged in layers, consisting of $10^2 - 10^4$ neurons. ANNs derive their power from arranging multiple layers, each of which successively processes the input data to extract increasingly specific features.

Although invented in the 1940s the field only started to gain mainstream traction this millennium [77]. Three main factors drove the uptake: better algorithms; powerful and affordable hardware; and the availability of large data sets. These advances hailed the advent of *deep learning*. From driverless cars to smart assistants to recommendation systems, deep learning and ANN models are now pervasive in society and shape our everyday life.

The key text on artificial neural networks is Ian Goodfellow's *Deep Learning* [78]. ANNs and deep learning have been reviewed in [77, 79] and its applications to bioinformatics in [80–84]. Many more reviews of these topics have been written, but we believe, subjectively, that the above references are the best and most relevant to this thesis.

Although beyond the scope of this thesis, we, as a society, must never lose sight of the ethical implications of using this technology [85–87].

To understand how ANNs function as a whole, one must first understand their constituent parts—of which there are many. Here, we introduce the components of ANNs before outlining various classes of ANN architectures. In Section 1.5 we review recent applications of ANNs within bioinformatics that allow biological data to be handled in unique ways that are not possible with any other type of machine learning model.

1.4.1.1 Architecture

ANNs are composed of two or more layers, each of which processes the input in some way. Deep learning uses ANNs that contain many *hidden* layers between the input and output layers (Fig. 1.7). Each layer of neurons consists of weights (internal states of neurons) and biases (trainable values that are added to each neuron's internal state, akin to b linear function constants in $y = ax + b$). The width of a layer refers to the number of neurons it

contains. Each layer takes an input x and applies a linear function

$$\mathbf{W} * x + b,$$

where $\mathbf{W}$ are the weights and b is the bias term. Remarkably, an ANN (multi-layer perceptron) with a single hidden layer of finite width is able to represent any mathematical function [88]! However, the required width of the hidden layer may be infeasibly large. Instead of using one very wide hidden layer, multiple smaller hidden layers tend to be stacked on top of each other.

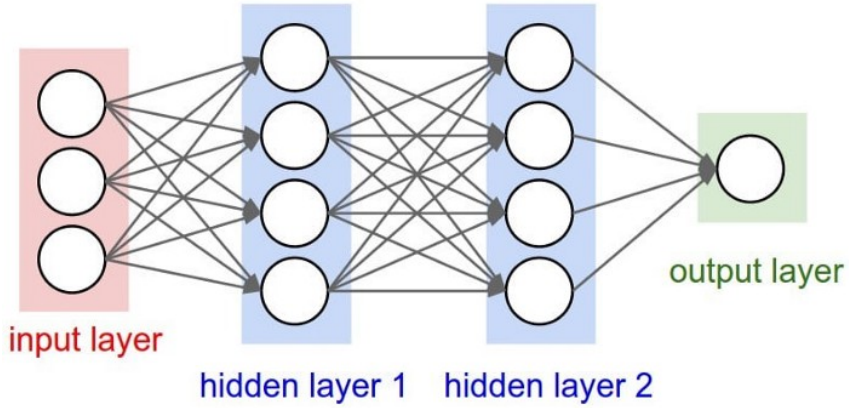


Figure 1.7: **General architecture of an artificial neural network.** The input layer is processed by two fully-connected hidden layers with the four neurons. The model outputs a scalar number from a single neuron. Figure taken from https://medium.com/towards-artificial-intelligence/artificial-neural-network-ship-crew-size-prediction-model-c04017c7b6fa.

Why is layer stacking so effective? Applying multiple linear functions in series, results in a linear function. Therefore, one might think that there is no added benefit to stacking layers. However, the power of deep ANNs comes not from their ability to learn linear functions, but from their ability to learn non-linear functions. To achieve non-linearity, non-linear activation functions are applied to the output of each neuron. The sigmoid (logistic) function

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

has long been the workhorse of ANNs. Sigmoid neurons, however, suffer from the vanishing gradient problem, which is caused by the difficulty of calculating gradients of large

positive, or negative, values. More recently, the rectified linear unit (ReLU)

$$\text{ReLU}(x) = \max(0, x)$$

and its variants mitigate this problem and perform better than sigmoid activations in deeper networks. ReLU also has the benefit of being cheap to compute.

1.4.1.2 Learning

In a feedforward ANN, information flows forward through the network, through the hidden layers, to produce the output (Fig. 1.7). ANNs frame the learning process as an optimisation problem, where an objective function is optimised. Typically the objective is to minimise the loss between the output and the ground truth. For example, in supervised machine learning, the difference between the predicted class probabilities and the ground truth class assignments is minimised.

Objective functions are optimised using gradient descent through a loss landscape. The topology of this landscape is dependent on the model's parameters. During the optimisation process, the model descends through the landscape until it reaches a minimum loss. To do so, the model needs to know the direction (gradient) to travel in to reduce its loss. The gradient of the objective function is computed by back-propagating the loss, backwards through the network [89]. An optimiser algorithm uses the gradient to update the model's parameters so that the loss of the model is reduced.

So far, we have only dealt with ideal cases where the model's loss always decreases. Learning is made tricky by the non-convex nature of many loss landscapes. In these, local minima, local maxima and saddle points are present where the gradient is 0. Therefore, these points provide no information about which direction to travel to reach the global minimum.

The partial derivative $\frac{\partial}{\partial \mathbf{x}_i} f(\mathbf{x})$ of a function $f(\mathbf{x})$ gives the direction of the function w.r.t. $\mathbf{x}_i$. By extension, the gradient of a vector $\nabla_{\mathbf{x}} f(\mathbf{x})$ is a vector containing containing all of the partial derivatives $\mathbf{x}_i$. ANNs use back-propagation to calculate gradients using the chain rule of differentiation.

Classically, ANNs have used the stochastic gradient descent (SGD) optimiser. Calculating the gradient using all training data is expensive, so SGD gets round this by estimating the gradient using a small sample of training examples. SGD introduces the concept of the learning rate ϵ that is multiplied with the estimate of the gradient $\hat{g}$ to update the

parameters $\theta = \theta - \epsilon\hat{g}$. The learning rate can be used to limit large updates to θ, which could occur for certain samples of training examples.

Optimisation of the objective function can become stuck at local minima and saddle points with SGD, which produces inferior models and increases training time. Contemporary optimisers use adaptive learning rates and momentum to train models rapidly by finding the correct direction to move in, whilst not getting stuck in local minima. Momentum encourages models to continue moving in the direction of past gradients, with an exponentially decaying impact. Adaptive learning rates use separate learning rates for parameters that are scaled according to their previous values. One such optimiser that combines both concepts, Adam [90], is particularly popular in the community.

1.4.1.3 Regularisation

Whilst it is easy to train a neural network to perform well on the training data, it can be hard to get good performance on unseen testing data. Regularisation can be applied to encourage models to learn general patterns and to not overfit on the training data. Overfitting is typically monitored on a validation set that is disjoint from the training and testing sets. A simple way to prevent overfitting is to monitor the objective function applied to the validation set. Training is stopped once the loss begins to increase on the validation set, due to overfitting on the training set.

Classical regularisation penalties, such as L1 and L2 loss can be added to loss functions. However, a popular, simple and cheap method of regularisation is dropout [91]. During training, neurons are dropped out at random with probability $1 - \rho$. Dropout is not applied at test time, so to ensure inputs are similar between training and testing, during training the activations of retained neurons are scaled by $\frac{1}{\rho}$.

Now we know how models can be trained, we next focus on different types of ANN architectures.

1.4.2 Encoder-decoders

Encoder-decoders are a general framework used by embedding methods [92]. In encoder-decoders, the encoder first transforms the input data to low-dimensional embeddings, followed by reconstruction of the original data by the decoder, using only the embeddings (Fig. 1.8)

$$\mathrm{DEC}(\mathrm{ENC}(\mathbf{X})) = \mathbf{X}' \approx \mathbf{X}$$

ENC and DEC are optimised such that $\text{ENC}(\mathbf{X}) = \mathbf{h}$ and $\text{DEC}(\mathbf{h}) = \mathbf{X}'$. Transitively, if the original data can be reconstructed by the encoder-decoder model, then the embeddings must contain all salient information in the original data. As such, the embeddings can be used as learnt features to train machine learning models.

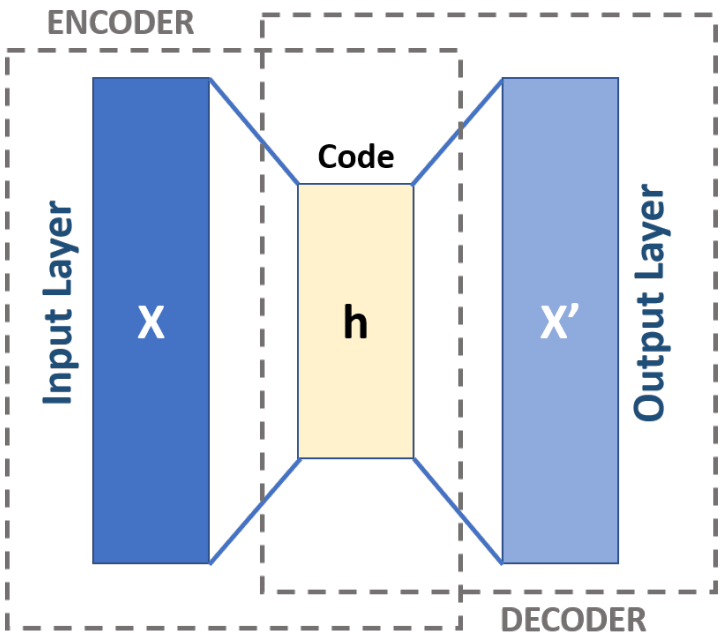


Figure 1.8: **Architecture of a general encoder-decoder model.** Figure taken from https://en.wikipedia.org/wiki/Autoencoder.

The encoder and decoder functions are learnt in an unsupervised way from the data using an optimisation process. In order to do this, a loss function must be defined to measure the difference between the original data and its reconstruction from the embeddings. The encoder and decoder functions and then optimised to minimise the reconstruction loss, and, concomitantly, the embeddings are improved.

Encoder-decoder models can be divided into direct-encoding and generalised methods [92]. Direct-encoding methods learn an embedding matrix $\mathbf{Z}$, where embeddings are simply

$$\text{ENC}(v_i) = \mathbf{Z}\mathbf{v}_i$$

where $\mathbf{v}_i$ is a one-hot indicator vector corresponding to element v_i.

We will encounter many applications of encoder-decoder neural networks in Section 1.5. In the next section, we explain one general application of the encoder-decoder framework to ANNs in the form of autoencoders.

1.4.3 Autoencoders

Autoencoders are unsupervised neural network encoder-decoder models that attempt to reconstruct their input $\mathbf{X}$ via a hidden representation $\mathbf{h}$ (Fig. 1.8). Typically, autoencoders are used for unsupervised feature learning, where $\mathbf{h}$ are a small number of informative features, where $|\mathbf{h}| << |\mathbf{X}|$, that are automatically extracted from $\mathbf{X}$. This process is also referred to as learning an embedding of $\mathbf{X}$. The features learnt in $\mathbf{h}$ can be used for supervised machine learning, distance estimation, dimensionality reduction, denoising data and modelling latent variables. We focus on applications of autoencoders in Section 1.4.2.

A typical autoencoder architecture might look like

$$\mathbf{X} \rightarrow \text{ENC} \rightarrow \mathbf{h} \rightarrow \text{DEC} \rightarrow \mathbf{X}'.$$

Here, an encoder function generates the hidden representation $\text{ENC}(\mathbf{X}) = h$ and a decoder function decodes $\mathbf{h}$ to reconstruct $\mathbf{X}$ as closely as possible $\text{DEC}(\mathbf{h}) = \mathbf{X}'$.

Autoencoders are trained to minimise the loss between $\mathbf{X}$ and $\mathbf{X}'$, $L(\mathbf{X}', \mathbf{X})$. The simplest way for the autoencoder to have a small loss is to learn the identity function $\mathbf{X} = \text{ENC}(\mathbf{X}) = \mathbf{h} = \text{DEC}(\mathbf{h}) = \mathbf{X}'$. However, as we shall see, this defeats the purpose of why one would want to use an autoencoder in the first place.

Restrictions are placed on the autoencoder so that the encoder function must actually learn from $\mathbf{X}$. The autoencoder is prevented from being able to reconstruct $\mathbf{X}$ perfectly and is forced to prioritise learning useful features in $\mathbf{h}$. As such, overfitting is not a problem when training autoencoders. Undercomplete autoencoders employ the simplest restriction, where $\mathbf{h}$ is made to have fewer dimensions than $\mathbf{X}$. Interestingly, when the encoder and decoder functions are linear and L is the mean squared error, an undercomplete autoencoder recapitulates PCA. With nonlinear encoder and decoder functions, autoencoders are able to learn nonlinear decompositions of $\mathbf{X}$ that are more powerful than PCA. In addition to restricting the number of neurons in $\mathbf{h}$, other restrictions can be applied, such as sparsity or noise. Sparse autoencoders apply an L1 sparsity penalty $\Omega(\mathbf{h})$ to the hidden layer, which encourages only important neurons in the hidden layer to fire. In denoising autoencoders, random noise is applied to the input $\tilde{\mathbf{X}}$ and the autoencoder is trained to minimise $L(\tilde{\mathbf{X}}', \mathbf{X})$. The autoencoder must learn to undo the noise and thus learn the true structure of $\mathbf{X}$.

Like many machine learning methods, autoencoders learn to map inputs to a low-

dimensional manifold. Variational autoencoders train latent variables to learn the structure of this manifold. These latent variables can be used to generate synthetic examples at some position on the manifold, and smoothly interpolate synthetic examples across the surface of the manifold.

1.4.4 Convolutional neural networks

Convolutional neural networks (CNNs) were introduced by Yann LeCun in his seminal 1989 work to recognise handwritten characters [93]. CNNs are ideally suited to grid-like data, such as sequences (1D) or images (2D). As the name implies, CNNs are ANNs that apply convolutional operations, as opposed to matrix multiplication, in at least one of their layers. CNNs gained notoriety when Alex Krizhevsky won the ImageNet object recognition challenge in 2012 using a deep CNN [94].

A convolutional layer is composed of a set of small filters that recognise features in the input. Each filter is only able to recognise a single feature, however this feature can be recognised at any position in the input. For example, convolution may be applied to a 5×5 window of pixels in an image. The architecture of a typical convolutional layer is often

$$\text{Conv} \rightarrow \text{Pooling} \rightarrow \text{Activation}$$

(pooling is explained below).

Convolutional layers provide three benefits to traditional neural network architectures [78]: parameter sharing, sparse interactions and translation equivariance. Parameter sharing occurs within a single filter, such that the filter's weights are applied to each portion of the input, instead of learning a separate weight for every coordinate. As such, a convolutional layer has a set of sparse interactions between its input and output governed by the size of the filter. Sparse interactions give rise to the receptive field of a convolutional layer. If a feature is recognised by a filter, it will be represented in its output. This output is equivariant to translation within the input: a feature occurring early in the input will be represented early in the output. If this same pattern occurred late in the input, its representation would be exactly the same but translated later in the output. It should be noted that convolution is not equivariant to scaling or rotational transformations.

For a convolutional layer with kernel of width k, the width of the output is shrunk by $k - 1$. The extent of downsampling is controlled by the depth of the convolutional layers and the widths of their kernels. To prevent shrinkage, zero padding can be applied before

convolution is applied by subsequent convolutional layers. In some cases, downsampling may be desirable to increase the computational efficiency of the model. Downsampling can also be achieved by applying convolution with a stride to skip some positions of the filter.

Pooling is applied to the output of a convolutional layer and calculates a statistic that summarises the outputs in a window. One type, max pooling, calculates the maximum value within a window of n inputs. For example, 4-max pooling of $\left[0.4, 0.7, 0.3, 0.5\right]$ produces 0.7. Pooling helps to make features invariant to translation, that is small changes in the input do not change the output. This property is desirable if the task is to recognise features in the input, regardless of their position. To improve a model's computational efficiency by reducing the number of parameters, pooling is often used to downsample an output by using fewer pooling units than output units.

1.4.5 Recurrent neural networks

Recurrent neural networks (RNNs) are a class of neural network that can be used to process sequences through repeated application of the network to each time step in the sequence. Like CNNs, RNNs employ parameter sharing, whereby the same parameters are used to process each time step. RNNs have been successfully applied to text, audio and sensor data. Whilst an in-depth introduction to RNNs is beyond the scope of this chapter, we introduce their salient details and limitations here.

Whilst RNNs are synonymous with sequence modelling [95], they do posses some significant drawbacks, namely sequential dependencies, vanishing/exploding gradients and poor memory, that we outline below. For these reasons, RNNs have fallen out of favour at the big tech companies, in favour of convolutional architectures, such as attention-based networks [96, 97]. A very recent class of CNN, temporal CNNs [95], are optimised for modelling sequences and outperform RNNs on a wide range of sequence modelling benchmarks.

- **Sequential dependencies**

  RNNs are applied sequentially to input sequences x of length τ from time point 1 to τ. The output of an RNN at time t is a function of $x^{(t)}$ and all previous time steps $x^{(1)}, ..., x^{(t^{-1})}$. Note that time steps need not actually correspond to *time*, but can instead correspond to *position* in the sequence. RNNs can even be used to predict future states of a sequence at times $t > \tau$, as is used in next word text prediction.

Despite the impressive results that have been obtained using RNNs, they can be challenging to train, due to the dependency of processing all time steps $1, ..., t^{-1}$ before processing t.

- **Vanishing/exploding gradients**

  To process a sequence, RNNs build a very deep computational graph, through which it is often difficult to calculate the gradient. This can lead to the problem of vanishing, or exploding, gradients during training, which make it difficult to know which direction to update a model's parameters—and in the case of exploding gradients can make parameters fluctuate wildly during training.

- **Poor memory**

  RNNs can learn dependencies between positions in sequences. However, learning long-term dependencies is challenging, due to the weights of dependencies decreasing in size exponentially with their distance. This means that short-term dependencies with larger weights will tend to dominate the learning process and that it will take a very long time to learn long-term dependencies. A vanilla RNN has been shown to struggle to learn dependencies in sequences of length 20 [98]. Variants of the RNN, such as the long short-term memory (LSTM) network [99] claim to increase this limit to $1{,}000$ time steps, however, in practice LSTMs are only able to handle sequences up to 250 in length.

1.4.6 Random forests

Decision trees are binary trees that segregate data into groups. Items begin at the root node, and traverse the tree towards the terminal nodes, with each internal decision node's predicate determining the which node the item will visit next. Because the global optimum partitioning of the data is NP-complete [100], the classification and regression trees (CART) algorithm uses a recursive greedy procedure to grow decision trees.

In the training phase, trees are grown by splitting the data according to the predicate that minimises the error at each internal node. A cost function is used to decide if a node is worth splitting, otherwise it will become a terminal node. Cost functions typically take into account the following criteria. If the node is split:

- Will the tree have reached the maximum depth?
- Will there be too few samples in either of the two nodes?
- Will the purity of the node be increased above some threshold?

Once a tree has been grown to its full length, it can pruned to prevent overfitting by reducing the number of terminal nodes without increasing the prediction error substantially.

In the prediction phase, each example begins at the root node and follows a path along the tree's branches—according to its feature values—until it reaches a terminal node. The label associated with the leaf node is assigned.

Whilst the CART algorithm is useful, it does have several problems. Firstly, the data may not be partitioned optimally, so the global minimum error will not be achieved. Secondly, CARTs are sensitive to the training data, and small changes to the training data can result in the growth of very different trees. Random forests (RFs) attempt to mitigate these two problems and help to reduce the variance of the model.

RFs are ensembles of CARTs. RFs use a technique called bootstrap aggregating, or 'bagging', to reduce the variance of the model. Each decision tree in an RF is trained on random subsets of the features and examples, from which bootstrap samples are drawn. Bootstraping is a resampling method that can be used to estimate a sampling distribution. Let X be some sample data drawn from an unknown distribution S. S can be estimated by n bootstrap samples, b. b_i is generated by sampling $|X|$ items from X with replacement. To increase the differences between each tree in an RF, a random subset of features can also be selected to be included in the training data for each tree. The proportion of samples and features to be included in each subset are hyperparameters, optimised during training.

The number of trees in the forest is not a hyperparameter that should be optimised [101]. Generally, the largest number of trees that can be trained appropriately on the available resources should be selected. Training and prediction of RFs is embarrassingly parallel because each tree is independent. However, for prediction, the added overhead associated with splitting the task across multiple processors does not typically outweigh applying each tree in serial on a single processor.

In the prediction phase, each tree in the RF has an equal vote. In the classification paradigm, RFs use the majority vote to predict the class of an example. In RF regression, the probability that an example is from some class is the proportion of trees that predicted that class

1.4.7 Support vector machines

Support vector machines (SVMs) are supervised machine learning models that classify data into one of two classes [102]. During training, the SVM learns a hyperplane in the training

data space that maximises the separation between the data points from the two classes. In the ideal case, data points from each class will be well separated, with a maximum margin between the hyperplane and the nearest data points from each class. The data points that lie on the margin on either side of the hyperplane are known as the support vectors. We discuss applications of SVMs in Sections 1.5.1.2 and 1.5.1.5.

SVMs are inherently linear classifiers, however SVMs are able to perform more complex, non-linear classification by use of the kernel trick. Instead of performing classification in the original data space, SVMs first map the data to a high-dimensional space, in which the two classes are more likely to be separable by a hyperplane. To do this, a kernel function is applied to the data, which calculates dot products between data points in a high-dimensional space, without actually mapping the data points to the high-dimensional space. Implicitly, kernel functions allow SVMs to produce a curved decision plane in the original feature space, which is actually a straight hyperplane in the kernel space. Kernel functions should be selected according to the particular task at hand, but one popular and flexible kernel, that produces curved decision boundaries, is the radial basis function kernel.

Real-world data is rarely ideal, so SVMs must make certain trade-offs when optimising the separating hyperplane between how important misclassifications and true classifications are. To do this, SVMs have a hyperparameter, the soft-margin penalty C, that controls the penalty applied to errors. Lower values of C produce boundaries with larger margins, which increase the risk of misclassifications but reduce the chance of overfitting. On the other hand, larger values of C produce decision boundaries with a small margin between the two classes, at the risk of overfitting on the training data. The hyperparameter should be optimised during model training, in addition to other hyperparameters that are specific to particular kernel functions. For example, the radial basis function kernel has the γ hyperparameter that controls the influence of the data points on the hyperplane, with high values increasing the locality of the influence and low values extending the range of influence.

1.5 Modern applications of machine learning to protein data

Since beginning my PhD, bioinformatics has witnessed an explosion of machine learning research based on ANNs—or 'deep learning' as it is sometimes, and often incorrectly, referred to. These topics have been reviewed in many publications, but we believe, subjec-

tively, that the following reviews are best [77–84]. The ensuing explosion has resulted in ANNs being used so widely in bioinformatics that it would be beyond the scope of any single piece of work to cover all topics. Instead, we focus here on the more relevant advances and applications to the task of protein function prediction and the related task of protein family prediction. Specifically, we focus on methods that allow machine learning to be applied directly to network and sequence data, without the need for feature engineering. These methods feel almost like magic and would have been inconceivable just a decade ago. Astute readers may notice some commonalities between the names of methods that we introduce here…

The timing of the advances in biological data generation and ANNs have coincided at an ideal time. High-throughput methods now generate unprecedented volumes of experimental data, that classical bioinformatics techniques struggle to process. ANNs are powerful statistical models that require large amounts of data and computational power to train. ANNs have already been proven to be a valuable tool in the bioinformatics toolbox. We conclude that we have set out along a path in which ANNs will permanently transform bioinformatics.

1.5.1 Protein networks

Graphs are ubiquitous data structures. Graphs have many important applications, including machine learning on graphs. For example, recommendation engines—such as those used at Netflix, Spotify and Amazon—predict shows, music and products for users. Here, we introduce a number of techniques that allow machine learning to be applied to graphs. All of the methods involve calculating a node embedding in the graph. Comprehensive reviews of the latest advances in this topic can be found in references [92, 103].

1.5.1.1 Graph embeddings

Embedding methods learn representations of nodes that encode structural information about the graph [92]. Embeddings are vectors that represent a set of latent features that are automatically learnt from the graph. The goal is to optimise the embedding, such that relationships in the embedding space recapitualate relationships in the graph space. Graph embeddings have been used previously for protein function prediction [65, 66].

The crucial advance of graph embedding methods is that the function that maps nodes to the embedding space is learnt from the data in an unsupervised way. Graph embedding methods are not domain-specific, so they can be applied to any graph.

Low-dimensional embeddings tend to be learnt, where on the order of $10^2 - 10^3$ latent dimensions are used. These embeddings are small enough such that they can be used to train off the shelf machine learning models. Alternatively embeddings can be used to calculate distances between nodes—in terms of the differences in their network contexts—with applications in clustering.

We encountered the encoder-decoder framework in Section 1.4.2. Many graph embedding methods are based on encoder-decoder models. The encoder first maps nodes to low-dimensional embeddings, followed by reconstruction of the original data by the decoder, using only the embeddings. Later, we introduce Mashup [65], node2vec [104] and DeepWalk [105] as examples of direct-encoding methods. This approach has a number of drawbacks, however. Firstly, no parameters are shared in the model and each node's embedding vector is learnt independently. Consequently, models are $\mathcal{O}(|V|)$. Parameter-sharing is a powerful regularisation approach. Secondly, models are static and embeddings can only be generated for nodes available at training time. This is limiting for temporal graphs that change over time and large graphs that cannot be stored in memory. Generalised methods, on the other hand, overcome these limitations by learning more complex encoder functions that share parameters and can be applied to nodes that were not available at training time. Autoencoders are a type of generalised encoder-decoder model, implemented using an unsupervised neural network. Below, we introduce deepNF [66] as an example of a generalised encoder-decoder method.

1.5.1.2 deepNF

deepNF [66] learns a set of highly informative features, by embedding proteins in a low-dimensional latent space that represents the context of each protein in a protein-interaction network. Conceptually, the multimodal deep autoencoder compresses large volumes of orthogonal information, encoded in multiple types of edges, into a small number of features. Although deepNF was devised for protein-interaction networks, the method is entirely flexible and can be applied to any network that contains multiple, orthogonal types of edges. deepNF was inspired by other (non-biological) approaches to applying deep learning to graphs, including Deep Neural Graph Representations [106] and Structural Deep Network Embeddings [107].

More formally, deepNF takes a multigraph G with n nodes and learns an embedding that represents the context of each node in this graph. Given k different types of edges in

G, an autoencoder is used to learn an encoder function that maps the adjacency matrix of G to a low l-dimensional embedding space that represents the context of each node across the k edge types. The encoder function is unary, embeddings are generated for each node. An alternative view of deepNF is that k input networks, each with n nodes, are used as input to a multimodal deep autoencoder (Fig. 1.9). Overall, the autoencoder maps an $[n \times n \times k]$ multigraph adjacency matrix to an $[n \times l]$ embedding matrix, where $l << n$ for any reasonably sized network.

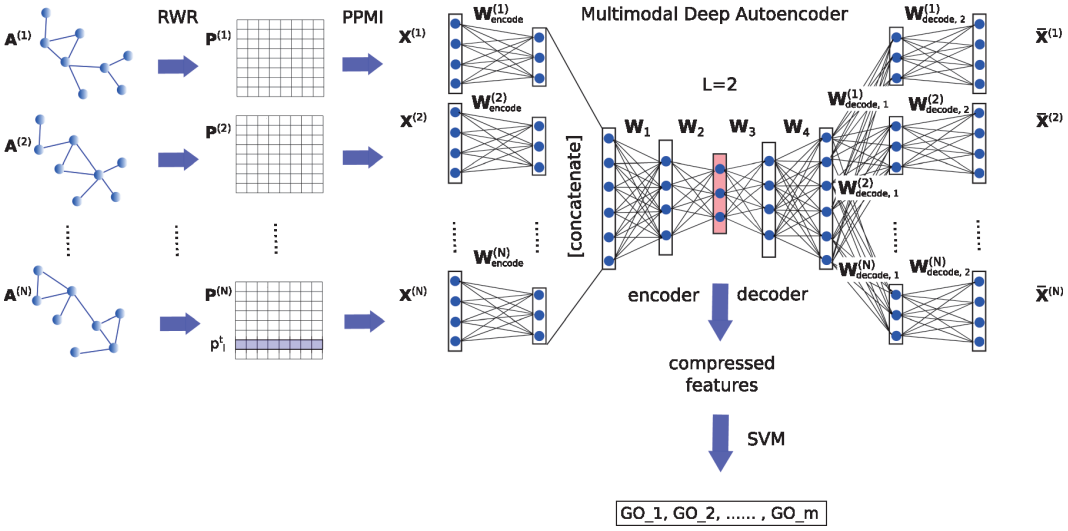


Figure 1.9: **deepNF overview.** Figure taken from [66].

Random walks with restart are applied to the adjacency matrices to calculate a node transition probability matrix. Local and medium-range topologies of the graphs are explored by the random walks. This, in turn, reduces the sparsity of the original adjacency matrix. The pointwise mutual information is then calculated that two nodes occur on the same random walk.

SVMs were trained to predict protein functions using the one-vs-rest multiclass strategy. Radial-basis function kernels were precalculated and cached to speed up training time. Terms were split into three levels according to how many proteins each term is annotated to. As one might expect, more common functions that are annotated to $101 - 300$ proteins were predicted better than rarer functions annotated to $11 - 30$ proteins.

deepNF, Deep Neural Graph Representations and Structural Deep Network Embeddings all suffer a number of limitations. Firstly, all require that the input dimension to the autoencoder is $|V|$. In the case of deepNF, the input is even larger, at $k|V|$ for k edge types.

Therefore, these methods cannot be applied to large networks with $> 10^5 - 10^6$ nodes, depending on memory resources. Secondly, models are fixed to the number of nodes in V at training time. However, new embeddings can be generated if the original adjacency matrix is rewired, for example under different developmental stages, cellular stresses or other changes in proteome regulation.

We use deepNF extensively in Chapter 4. deepNF took much inspiration from another graph embedding method, Mashup [65], which we introduce in the next section.

1.5.1.3 Mashup

Mashup [65] is a similar method to deepNF, in that it learns a low-dimensional embedding of nodes across multiple input networks. Like deepNF, Mashup was developed for protein networks, but the method is not domain-specific and can be applied to model any multi-graph problems. However, unlike deepNF, Mashup is a direct-encoding method, cannot be generalised to new nodes and does not share parameters between nodes.

Mashup first calculates a node transition probability matrix using random walks with restart. Low-dimensional embeddings are then calculated by applying a novel dimensionality reduction method. Networks are high-dimensional, incomplete and noisy. We wish to transform the original matrix into a low-dimensional matrix that explains the variance of the original matrix, similar to PCA. Mashup achieves such a dimensionality reduction by framing the process as an optimisation problem. Each node i is represented using two vectors: x_i for features of the node and w_i that captures the context of the node in the topology of the network. (If x_i and w_j are close in direction and have large inner product, then node j will be visited often on random walks beginning at node i.) If these vectors x and w do indeed capture topological features of the network, then they can be used to identify similar nodes in the network. Mashup optimises the x and w vectors by minimising the Kullback–Leibler divergence between the observed transition probabilities s and the predicted probabilities $\hat{s}$. This optimisation procedure can be extended to multiple networks, allowing node contexts across heterogenous types of edges to be integrated. The trick is that each of the k networks has its own w_i^k context vector, but node vectors x_i are shared across all networks. The objective function is jointly optimised across all networks. In so doing, latent features of nodes are learnt in an unsupervised way and are captured in the node vectors x, which can be used to train machine learning methods.

Node embeddings were successfully applied to a number of biological problems. In all

cases, Mashup was benchmarked against a state-of-the-art method and achieved a higher performance.

- Predict protein function, benchmarked against GeneMANIA [108].
- Reconstruct Gene Ontology, benchmarked against NeXO [109].
- Predict genetic interactions, benchmarked against Ontotype [110].
- Predict drug efficacy, benchmarked against a synthetic lethality predictor for cancer drugs [111].

1.5.1.4 node2vec and DeepWalk

Random walks are also used by two similar methods, node2vec [104] and DeepWalk [105], to explore the topology of a network. Both methods learn a pairwise encoder that encodes properties of random walks on the graph between a pair of nodes i and j. Consequently, embeddings are stochastic and asymmetric, i.e. they are direction-specific from i to j. These methods frame network embedding as a problem in the style of natural language processing, where nodes are words, the set of random walks is a corpus of text and paths are sentences sampled from the corpus.

DeepWalk is based on a random walk model that uses the edge weights to decide each node-to-node transition. node2vec, on the other hand, implements a more advanced random walk model that is parametric and can be biased to perform more depth-first or bredth-first searches.

node2vec was extended to handle multiple networks, representing different types of edges, in OhmNet [112]. Whilst no parameters were shared, a regularisation penalty was applied that linked the embeddings of a node across each of the k networks.

We focus here on methods that employ random walks on graph nodes. A number of methods exist that are not based on random walks but are conceptually similar to node2vec and DeepWalk, including LINE [113] and HARP [114].

1.5.1.5 Graph kernels

Graph kernels can be calculated by applying a kernel function to all node pairs in the graph [115]. The kernel value for each pair of nodes recapitulates the context of these nodes in the graph and the kernel represents the overall topology of the graph. Graph kernels of functional association networks, such as protein-interaction networks, can be used for protein function prediction [116, 117].

Firstly, kernels can be queried directly using a seed set of proteins that are known

to have some function [117]. Under the guilt by association framework, the remaining proteins in the kernel can be ranked by their similarity to the seed set. Highly-ranked proteins are likely to have the function, and vice versa for low-ranked proteins.

Secondly, kernels can be used for data fusion because a combination of kernels is also a kernel [102]. As such, heterogenous information across multiple networks can be fused by combining their graph kernels. This approach was taken by Hériché et al. to fuse a range of human gene and protein association networks, in order to predict novel genes involved in chromatin condensation [116]. Nine genes known to be involved in this process were used as seeds to rank the remainder of the genome. An RNAi screen of the 100 best-ranked genes identified 32 that caused defective chromatin condensation when knocked down. Hit rates in RNAi screens are notoriously low, so these results correspond to an order of magnitude improvement on the median hit rate in mammalian cells [118].

Third and finally, kernels can be used directly in kernel-based machine learning models (Section 1.4.7). SVMs are an obvious type of kernel-based model, but many other types of models exist [102], including kernel partial least squares regression, kernel principal component analysis and kernel Fisher linear discriminant analysis. Often the radial basis function kernel is used in SVMs, but, if desired, tailored kernel functions can be used and the resulting kernel used directly in the SVM. For example, Lehtinen et al. [117] used a commute time kernel of the STRING [119] network to predict protein function using kernel partial least squares regression. This method outperformed GeneMANIA [120], the best performing method at the time.

Principal component analysis is the eigenvalue decomposition of a positive semi-definite covariance matrix. Adjacency matrices are not positive semi-definite, but the Laplacian matrix and graph kernels are. Principal component analysis has also been applied to commute time graph kernels [121]. In so doing, spectral clustering [122] of the principal components therefore has a tangible interpretation in terms of random walks on the graph nodes [121], where clusters are sets of nodes in the graph with high modularity.

1.5.2 Protein sequences

One of the most powerful features of ANNs in computational biology is the ability for models to be applied directly to DNA, RNA and protein sequences [80]. Classical machine learning methods—such as decision trees and support vector machines—require manual feature engineering to extract information from sequences manually. Examples

of hand-engineered features include k-mer frequencies, torsion angles, presence of secondary structural elements and distance from a protein functional site [123]. Often motif information is not captured [124]. ANNs do not require features to be hand-engineered. Instead, models can learn to extract features directly from the data. In the case of biological sequence data, models can detect sequence motifs by learning correlations between positions in the sequence, similar to, but more powerful than, HMMs [124, 125].

Biological sequences are vectors of categorical features from an alphabet Σ. Neural networks require numerical inputs, so sequences are one-hot encoded. For example the DNA 4-mer ATCG could be one-hot encoded as:

$$\begin{bmatrix} \text{A} \\ \text{C} \\ \text{G} \\ \text{T} \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

As such, one-hot encoded sequences can be thought of as $[n \times 1]$ images with Σ colour channels.

Most ANNs—with the exception of certain convolutional and recurrent architectures—require fixed-width inputs. Autoencoders require fixed-width inputs, so that the model can reconstruct inputs from encodings. Collections of variable-width sequences can be converted into fixed-width inputs by padding one-hot encodings with zeros [125] or a uniform distribution over Σ [126].

Studies using genomic sequences tend to divide sequences into shorter, more manageable 600 bp [127] or $1{,}000$ bp sequences [128–130] centred on a region of interest. Protein sequences are much shorter than genomic sequences, so entire protein sequences tend to be used.

One particularly interesting application of ANNs to biological sequences is that of embedding sequences in a low-dimensional latent space. In the same way as graph embeddings introduced in Section 1.5.1.1, these embeddings can be used as features to train off the shelf machine learning models. In the next section, we describe how sequence embedding methods work and introduce some of the ways they have been applied to sequence-based prediction problems in bioinformatics.

1.5.2.1 Sequence embeddings

The field of natural language processing has developed an array of machine learning methods to learn from text. One nascent approach is that of text embedding. These methods embed variable-length sequences into fixed-length, low-dimensional vectors. The first method to implement this approach was word2vec [131], which embedded words and phrases. word2vec inspired many other approaches, including doc2vec [132], for longer sequences of sentences, paragraphs and documents.

Recent natural language processing methods are able to capture information about word order and semantics. Examples include the continuous bag of words model, where a set of context words are used to predict a target word, and the skip-gram model, where a target word is used to predict the context word. Using these methods in embedding methods allows the embedding space to capture information about word order and semantics in the original text space. Thus, 'dog' and 'cat' will be closer in the embedding space than 'dog' and 'car', despite differing by only one letter. Word embeddings can be used to answer algebraic questions [132], such as

$$\overrightarrow{\text{king}} - \overrightarrow{\text{man}} + \overrightarrow{\text{woman}} = \overrightarrow{\text{queen}}.$$

Embeddings allow off the shelf machine learning models, that require a fixed number of features, to be trained on any set of sequences. Alternatively, vectors can be directly compared using distance metrics in vector space, such as the cosine distance.

Sequence embeddings can also be used for alignment-free sequence comparison [133]. Sequence alignment is expensive, particularly when performing pairwise alignments of large sets of sequences. Other alignment-free sequence comparison methods, such as those based on hashing, provide an approximate measure of the similarity between two sequences, without direct comparison of the sequences. These methods represent sequences as a low-dimensional vector by selecting m unique k-mers from each sequence. Each k-mer is hashed to a number (hash value), which is used to determine whether to select this k-mer to represent the sequence. MinHash is one hashing method that represents sequences using the m numerically smallest unique hash values obtained by hashing all k-mers in a sequence [134–137]. Sequences can be compared rapidly by calculating an

approximate distance as the pairwise Jaccard index,

$$J(X, Y) = \frac{|X \cap Y|}{|X \cup Y|},$$

between sets of hash values X and Y. Although hashing techniques are incredibly efficient and powerful, sets of k-mers do not retain semantic information about the context of k-mers in the sequence. The sequence embedding methods that we introduce below are able to represent sequences using a low-dimensional vector that does retain semantic information.

Convolutional architectures have permitted networks to be trained on raw data, such as DNA and protein sequences [80]. Hand-engineered features no longer need to be calculated from sequences; instead, the CNN learns to extract (non-linear) features from sequences automatically. Not only does this save time, but the extracted features are high-quality and lead to greater performance [80].

ANN text embedding methods are based on the encoder-decoder model that was introduced in Section 1.4.2. However, whereas network embedding methods use autoencoders to learn embeddings in an unsupervised manner, in sequence embedding, natural language processing methods are used to generate embeddings from unlabelled data. Whilst sequence embedding methods are, overall, unsupervised, the learning objective is posed as a supervised classification problem. One interesting early application of this approach was in Semantic Hashing [138] for document classification, however, here we focus on more recent methods.

1.5.2.2 word2vec

word2vec [131] takes a corpus of text, composed of words from word set $\mathcal{V}$, and generates n-dimensional embeddings for each word. A neural network is constructed, consisting of a V-dimensional input layer, where $V = |\mathcal{V}|$, an N-dimensional hidden layer and a V-dimensional output layer (Fig. 1.10; we use the notation from this figure in the remainder of this section). The architecture is intentionally shallow to allow it to be trained efficiently on large corpuses, where $V > 10^9$. This architecture is reminiscent of an autoencoder with a single hidden layer, but the training objective is different.

word2vec can be run in two modes: continuous bag of words or skip-gram. We use the continuous bag of words mode as an example. Here, C context words x_k are fed to the network as one-hot V-dimensional vectors and are processed by a word matrix $\mathbf{W}$.

Neurons in the hidden layer h are trained, using the average of the C context word outputs, to predict the target word y_j. Embeddings are processed by the $\mathbf{W}'$ matrix to generate outputs. Outputs are converted to probabilities using the softmax function. The network is optimised by reducing the loss between the target word y_j and the predicted words y.

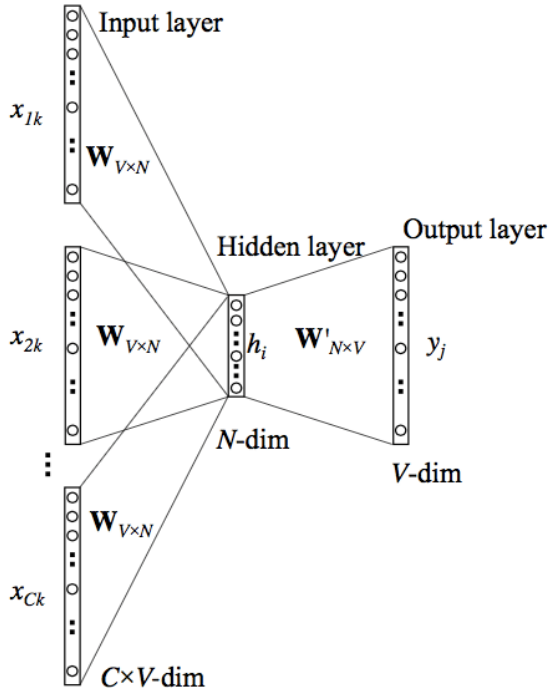


Figure 1.10: **word2vec neural network architecture.** The continuous bag of words mode is shown. Figure taken from http://www.stokastik.in/understanding-word-vectors-and-word2vec/.

1.5.2.3 doc2vec

doc2vec [132] is based on word2vec and generates n-dimensional embeddings for each paragraph P. One-hot V-dimensional vectors of words are shared across all paragraphs and are processed by the word weight matrix $\mathbf{W}$ (Fig. 1.11). In addition to being trained on context words, models are simultaneously trained on paragraph IDs. The paragraph ID is shared across all contexts of a paragraph and is processed by the paragraph weight matrix $\mathbf{D}$. The paragraph ID can be thought of as an additional word that links the context words to particular paragraphs. Therefore, the model is encouraged to learn the semantics of individual words and how they shape the meaning of the paragraphs they occur in. Each paragraph is mapped to an N-dimensional vector and each word is mapped to an M-dimensional vector. These vectors are concatenated and used to predict the target word in the same way as word2vec. At each step of the training process, C context words are

randomly sampled from P and are used to predict the target word.

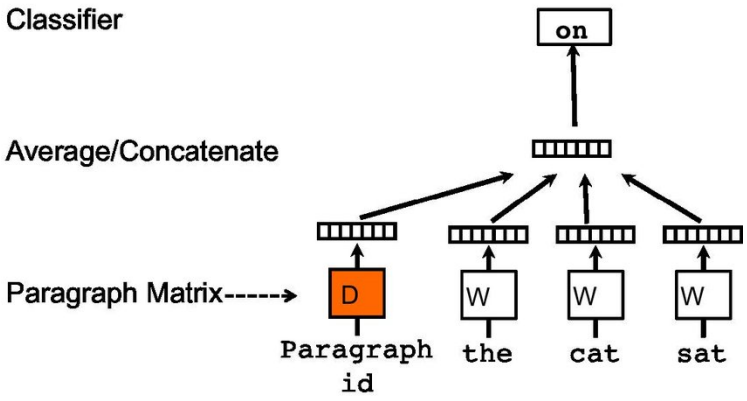


Figure 1.11: **doc2vec neural network architecture.** The continuous bag of words mode is shown. Figure taken from [132].

word2vec and doc2vec have been applied in many different ways to biological sequences. Below, we focus on applications to protein sequences. These methods have also been applied to nucleic acid sequences [139, 140].

1.5.2.4 BioVec a.k.a. ProtVec

BioVec a.k.a. ProtVec [139] is a sequence embedding method for biological sequences, based on word2vec. Sequences are treated as sentences from a corpus, where k-mers of the sequences are treated as words. BioVec embeds sequences according to k-mer (word) contexts. Sequences were embedded in a 100-dimensional space, where the sequence embedding is the summation of embeddings for all 3-mers in the sequence. BioVec found that 3-mers were the best length.

The embedding space was shown to encode biophysical properties of the 3-mers—for example, mass, polarity, hydrophobicity and charge—where 3-mers with similar biophysical properties had similar embeddings when projected into a 2-dimensional space using tSNE.

One-vs-rest SVMs were trained on embeddings to predict Pfam families for all of Swiss-Prot, using sequences from a family as positive examples and randomly selected negative examples from all other families. BioVec achieved 93% accuracy on this binary classification task. However, the setup of this experiment is far from how a protein family classifier would be used in real life. Pfam contains 7,027 families, so the negative set is likely to contain proteins that are significantly different to the positive set—making the binary classification task easy. Ideally, protein family classification needs to be performed

using a multiclass prediction strategy, rather than one-vs-rest. Protein family prediction using ANNs is covered in more detail in Section 1.5.3.

The 100-dimensional 3-mer embeddings were used in SDN2GO [141] to predict GO terms for proteins. SDN2GO is a multimodal ANN model that used network and InterPro protein family information in addition to protein sequences. The BioVec embeddings were processed by two convolutional layers followed by one fully connected layer, before being combined with network and protein family layer outputs to make the final function predictions.

ProtVecX [142] is a recent extension to ProtVec, which allows variable-length motifs to be extracted from protein sequences using the byte-pair encoding compression algorithm, which has gained in popularity in the NLP field. The standard ProtVec method is then used to generate embeddings of the extracted motifs.

1.5.2.5 seq2vec

seq2vec [143] is a sequence embedding method for biological sequences, based on doc2vec. seq2vec improves upon BioVec because the overall context of k-mers (words) in sequences (documents) are learnt, as well as the semantics of individual k-mers. Sequences were embedded using 3-mers into a 250-dimensional space.

One-vs-rest SVMs were trained on embeddings to predict Pfam families for all of Swiss-Prot using with 95% accuracy. Multiclass prediction was also tested for seq2vec using a multiclass SVM trained using the one-vs-one strategy, where $\binom{n}{2}$ models are trained to predict all pairs of classes. Here, the accuracy drops to 81% for seq2vec and 77% for BioVec. However, only the largest 25 Pfam families, which is $< 1\%$ of all families. To be useful for protein family classification in the wild, these methods will need to be able to classify all CATH or Pfam families simultaneously (see Section 1.5.3 for more details on protein family classification). It would be interesting to see how well an MLP performs that is trained to predict a larger number of families simultaneously.

Notably, seq2vec and BioVec were benchmarked against BLAST. Whilst seq2vec consistently performed significantly better than BioVec, BLAST outperformed seq2vec. However, BLAST is one of the most highly engineered pieces of bioinformatics software, so it is natural to expect it to outperform nascent embedding-based methods. At least for now!

1.5.2.6 dom2vec

dom2vec [144] is a sequence embedding method for protein sequences that learns to embed protein domains. dom2vec is based on word2vec [131] using the continuous bag of words and skip-gram strategies. Philosophically, BioVec and seq2vec treat amino acids as words and proteins as sentences, whereas, dom2vec treats domains as words and multi-domain architectures as sentences. InterPro domain MDAs for all UniProt sequences were used as input to a word2vec model dom2vec was benchmarked against ProtVec [139], which is also based on word2vec, and SeqVec, achieving competitive results. Extensive benchmarks were conducted for predicting EC classes, molecular function GO terms, InterPro domain hierarchies and SCOPe secondary structure classes.

1.5.2.7 SeqVec

SeqVec [145] is a sequence embedding method that is not based on word2vec. Instead, SeqVec is based on ELMo [146], an NLP model that predicts the next word in a sequence, given all previous words in the sequence. ELMo is comprised of a CharCNN, followed by two bi-directional LSTM RNN layers. The CharCNN learns a context-insensitive vector representation of each character, in this case each amino acid. The bi-directional LSTM layers take the CharCNN embedding and introduce contextual information. The forward and backward passes of the LSTM layers are trained independently to avoid the backward and foward passes leaking information to each other. SeqVec used a 28 letter alphabet to represent the standard amino acid code, ambiguous residues and special tokens to mark the beginning and end of sequences and padding. SeqVec generates a $1{,}024$D embedding vector from the summation of the $1{,}024$D outputs of the three layers at the C-terminus of the sequence. SeqVec was trained on 3×10^7 protein sequences from UniRef50, containing 9×10^9 residues. Each residue is a unique sequence context, thus was treated as a separate token in the NLP sense. SeqVec achieved competitive performance in benchmarking, but had faster run times than the competing methods.

1.5.2.8 UniRep

UniRep [147] is another LSTM-based sequence embedding method based on an NLP language model. UniRep and SeqVec were published two months apart, so neither benchmark the other. The model is trained on protein sequences to predict the next residue in the sequence. Like SeqVec, an amino acid character embedding is used as input to an LSTM. Each protein sequence is represented using a $1{,}900$D embedding vector. In order to cap-

ture long-range and higher-order dependencies between residues, the embedding was the average of the LSTM layer's hidden state for each amino acid in the protein sequence. Conversely, SeqVec just use the hidden state of the LSTM at the final residue in the sequence.

Whilst both SeqVec and UniRep are no doubt powerful models, they both took approximately three weeks to train for one epoch. This is in stark contrast to word2vec, which is designed for fast run time on large corpuses.

1.5.3 Protein families

Recently, ANNs have been applied to the problem of protein family classification. Protein families are defined by clustering protein sequences into sets of homologous sequences that share sufficiently high sequence identity. Until recently, HMMs have been trained to model the sequence diversity of protein families. HMMs can be applied to new sequences, that were not used to train the HMM, to assign them to one or more family, dependent on obtaining a sufficiently high match score. Sequence-based ANNs, such as CNNs [125, 148–150], RNNs [151] and natural language processing models [150], are well-suited to protein family classification because they can be trained directly on protein sequences. Models are trained to learn a mapping from protein sequence to a vector of protein families P. ANNs are not only able to learn a good mapping function [125, 148, 149], but sequences can be classified into protein families much faster than HMMs [125, 149], due to the fast inference time of ANNs and parallel execution on GPUs. Below, we introduce a selection of the best methods for protein family classification using ANNs.

1.5.3.1 DeepFam

DeepFam [125] classifies protein sequences into families of proteins. Eight convolutional filters of lengths $\{8, 12, 16, 20, 24, 28, 32, 36\}$ were applied to each protein one-hot encoded protein sequence. The model is able to handle variable-length sequences by 1-max pooling the convolutional stage, whereby the maximum activation from each of the eight filters is taken. This fixed-width vector is used as input to a single hidden layer used for classification. A variant of one-hot encoding is used that accounts for pairs of amino acids with similar structure and chemistry. When one-hot encoding protein x to X, if residue x_i is in one of three pairs of residues, 0.5 is entered in the jth position corresponding to

each residue of the pair,

$$X_{ij} = \begin{cases} 1 & \text{if } x_i = j\text{th residue} \\ 0.5 & \text{if } x_i = \alpha \text{ and } j\text{th residue} \in \{\mathrm{D}, \mathrm{N}\} \\ & \text{or } x_i = \beta \text{ and } j\text{th residue} \in \{\mathrm{E}, \mathrm{Q}\} \\ & \text{or } x_i = \gamma \text{ and } j\text{th residue} \in \{\mathrm{I}, \mathrm{L}\} \\ 0 & \text{otherwise.} \end{cases}$$

In the future, it would be useful if model performance was compared between one-hot encoded data using an alphabet of the 20 standard amino acids Σ_{20} versus a reduced alphabet that merges amino acids with similar properties $\Sigma_{N<20}$. The method was benchmarked using two protein family databases that were not built using HMMs: COGs and a manually curated GPCR set.

1.5.3.2 DeepSF

DeepSF [148] predicts the structural fold of a protein using its sequence as input. The input data for a protein sequence of length L consists of the sequence one-hot encoded (20D), a position-specific scoring matrix generated by PSI-BLAST (20D), predicted secondary structure (α-helical, β-sheet or coil; 3D) and predicted solvent accessibility (exposed or buried; 2D) to yield an $[L \times 45]$ matrix. DeepSF begins with a very deep CNN with 10 convolutional layers each with 10 filters $10 \times (L \times 2)$. The model is able to handle variable-width inputs by applying k-max pooling to the output of the 10^{th} convolutional layer. The 30 largest activations from each of the 20 $L \times 1$ feature maps are taken and flattened into a 600 neuron dense layer. An MLP maps the activations from the convolutional layers to 1,195 protein folds. To train on variable-length sequences, proteins were binned into mini-batches that contain proteins within a range of lengths and proteins within each mini-batch were zero padded to the same length.

1.5.3.3 ProtCNN

ProtCNN [149] used a CNN to classify protein sequences into protein families by taking the majority vote from ProtENN, an ensemble of 13 ProtCNN models. ProtCNN alone had higher error rates than BLASTp- or HMM-based classifications, but these classical methods were consistently beaten by ProtENN.

1,100D embeddings were calculated for representative sequences from protein fami-

lies. Nearest neighbour classification was used to assign held out sequences to families by calculating their embedding vector, followed by calculating cosine distances to all protein families and finding the most similar family. ProtCNN was also able to embed sequences from completely held out families into a similar region of embedding space, with small cosine distances. Consequently, this demonstrates that the CNN was able to learn general features of protein sequences, rather than merely memorising the training data.

Similar to DeepSequence [152] (Section 1.5.4.3), ProtCNN was able to learn a substitution matrix that is very similar to the BLOSUM62 matrix, but using the cosine similarity between 5D vectors centred on the residue of interest.

1.5.3.4 Multimodal deep representation learning

Multimodal deep representation learning [150] was used to classify protein sequences into families using sequence and PPI networks. The model consists of two phases, an unsupervised phase that learns features from the data, followed by a supervised learning phase that predicts protein families.

For the unsupervised phase, a stacked autoencoder is used to learn embeddings of pairs of protein sequences. Also a technique from the field of natural language processing is used to encode the context of each node in a PPI network. A continuous bag of words model is applied to metapaths along nodes in the PPI network. In a continuous bag of words model, the task is to predict a target word given some other words for context. Metapaths are a type of random walk that model the likelihood of a path subsequently visiting some node v_i, given all preceding nodes in the path [153]. Metapaths imply that nodes that tend to occur close to each other in a path have some meaningful relationship in the network. The task, when applying a continuous bag of words model to metapaths, is to predict a target node v_i given a set of context nodes $\mathbf{v}$, where $v_i \notin \mathbf{v}$.

In the supervised phase, sequence embeddings and network context information from pairs of proteins are combined to predict whether two proteins are related. This relation can either be that two proteins are in the same protein family, or that two proteins interact physically. Protein families were predicted from two databases: Database of Interacting Proteins [154] and Human Protein Reference Database [155]. It is unclear why the authors chose to use these databases, which were last updated in 2004 and 2009, respectively.

1.5.4 Protein function

Predicting function from sequence is the holy grail for protein function prediction.

Until recently, this was simply impossible. Extensive amounts of features engineering were required to extract low-dimensional, fixed-width representations of the information within protein sequences. These features were reasonably predictive of protein function, provided sufficient training examples were available. However, this feature engineering is tedious. Ideally, we want these features to be extracted from the sequence automatically. Over the past couple of years, this has been made possible by applying ANN models directly to protein sequences to predict protein function. Many of these applications have already been covered in Section 1.5.2.1. Here, we introduce methods that are not based on encoder-decoder embedding methods. First, we would like to highlight two studies without introducing them in detail:

- DeepText2GO incorporated textmining information alongside protein sequence information to predict GO term annotations [156].
- Whilst not utilising protein sequence information directly, an interesting approach was taken in [157] to predict many GO terms simultaneously using multi-task learning. Here, a set of GO terms were predicted using some common layers, shared between all GO terms, and smaller layers that are specific to each GO term being predicted. As such, the models are able to learn common features about protein sequences in the common layer, and GO term-specific information in the individual sets of layers for each GO term.

Next, we introduce methods that predict protein function from sequence.

1.5.4.1 DeepGO

DeepGO [158] predicts GO term annotations for proteins using a model that is aware of the graphical DAG structure of the GO. The model takes protein sequence as input and maps overlapping 3-mer sequences to indexes in a vocabulary of all 20^3 possible k-mers of length 3. Sequences were fixed at $1{,}002$ amino acids in length, corresponding to $1{,}000$ 3-mers. Longer sequences were ignored and shorter sequences were padded with zeros. Sequences are then embedded in a 128D space, where each 3-mer index is represented by a vector of 128 numbers. This is an interesting approach, but one wonders why a sequence embedding method like seq2vec was not employed. Network data was also included in the model by generating 256D embeddings of nodes within a cross-species knowledge-graph [159].

The model learns dependencies between terms in the GO by representing each GO

term with its own small neural network. Each term consists of a single fully-connected layer with sigmoid activations. Top-level terms in the GO DAG take as input the concatenated sequence and network information. Terms that have child terms in the ontology feed the output of their fully-connected layer to the fully-connected layers of any child terms. The output of the term layers are fed to an output vector that predicts GO terms in a way that is aware of the correlations and dependencies between terms in the ontology.

However, DeepGO was unable to overcome the classical problems with GO term prediction. Firstly, the problem is high-dimensional in the number of GO terms that exist. Secondly, GO terms that are deep in the ontology, and describe specific functions, are annotated to only a few proteins. So, instead of predicting all GO terms, a subset of terms that are annotated to many proteins were selected. Terms were selected if they are annotated to 250 proteins for biological process and 50 proteins for molecular function and cellular component. This resulted in 932 terms for biological process, 589 for molecular function and 436 for cellular component.

DeepGOPlus [160] is the prototypical ANN model for protein function prediction. The model is simple and intuitive, taking protein sequences as input and predicting GO terms as output. DeepGO was modified in three ways to create DeepGOPlus. Firstly, the 3-mer embedding stage is replaced by a one-hot encoding of the sequence, thus removing 128×8000 parameters from the model and reducing the chance of overfitting. Furthermore, this architecture allowed DeepGOPlus to be applied to any length sequences. Secondly, the CNN unit was converted to a deep CNN unit, consisting of stacked convolutional layers. Thirdly, network information was not used because network information is unavailable for most known proteins. Finally, GO terms were predicted using a flat fully-connected layer, rather than the hierarchical set of layers used in DeepGO. DeepGOPlus would have come in first or second place for the three GO ontologies in CAFA 3 [74].

It is unclear how many GO terms DeepGOPlus can predict simultaneously. In the paper, $5{,}520$ GO terms were predicted—from all three ontologies in the CAFA 3 data sets. The model has state-of-the-art performance, so it is reasonable to expect that the number of predicted GO terms could be increased to include a larger fraction of all possible GO terms.

1.5.4.2 ProLanGO

Neural machine translation models, such as Google Translate, allow text to be translated between arbitrary pairs of languages using ANNs. ProLanGO [161] is a neural machine translation model for GO term prediction. In this model, protein sequences and GO term annotations are treated as languages and a mapping function is learnt by an ANN to translate between the semantics of particular protein sequences and their equivalent GO term semantics. The model is an RNN, composed of LSTM units.

The protein sequence language was constructed of words that are all k-mers of length 3, 4 or 5 that occur in UniProt > 1000 times. The GO term language is constructed by assigning each GO term to a unique four letter code word in base 26. The four letter code is the index of the term from a depth-first search of the GO DAG. For example, there are $28{,}768$ terms in the biological process ontology, so the root node of the ontology, GO:0008150, is the 28768^{th} term to be visited in the depth-first search, which corresponds to BQKZ in four letter code.

1.5.4.3 DeepSequence

Autoencoders have only recently been applied to biological sequences. DeepSequence [152] predicts the effects of point mutations using a variational autoencoder in a Bayesian framework. Most mutation effect predictors do not take into account long-range interactions between residues. DeepSequence improves upon this by learning a set of latent variables that model pointwise dependencies, pairwise interactions and long-range interactions between residues. The latent variables can be used to predict the effects of particular point mutations in the amino acid sequence using the log likelihood ratio for the mutant residue w.r.t. the wild type residue, $\log \frac{p(x_{\text{mutant}})}{p(x_{\text{wild type}})}$. DeepSequence was able to capture correlations between residues that recapitulate the physiochemical properties of amino acids. This was used to generate a substitution matrix that correlated well with BLOSUM62 (Spearman $\rho = 0.83$) and may indeed be better than BLOSUM62. A similar approach was taken by Sinai et al. [162], who also show how the probabilities of residues across the entire protein sequence change in their model when a single residue is mutated *in silico*.

1.6 Contributions of this thesis

This thesis documents four protein function prediction research projects, whose contributions are outlined below.

In Chapter 2, we study how the brain proteome is affected by Alzheimer's disease and identify new genes involved in the progression of disease. We used an inducible *Drosophila melanogaster* model that expresses Aβ42, a variant of the amyloid beta gene associated with an aggressive form of Alzheimer's disease. Abundances of brain proteins were tracked over time using label-free quantitative mass spectrometry. We identified 228 proteins that were significantly altered by Aβ42 accumulation and were enriched for AD-associated processes. Network analyses further revealed that these proteins have distinct hub and bottleneck properties in the brain protein interaction network, suggesting that several may have significant effects on brain function.

Second, in Chapter 3, we predict novel plastic hydrolase enzymes in a large data set of 1.1 billion protein sequences from metagenomes using the CATH database. We mapped a naturally-evolved plastic hydrolase from *Ideonella sakaiensis* to the alpha/beta hydrolase CATH superfamily and FunFams. By scanning the metagenomic proteins against HMMs of these families, we identified 500,000 putative sequences that may be able to hydrolyse plastics, which we analysed further using associated metadata. Motivated by the size of the metagenomic protein data set, we developed FRAN, a divide-and-conquer algorithm that is able to generate FunFams on arbitrarily large sequence data sets.

Third, in Chapter 4, we perform feature learning from protein networks using a neural network that generates embeddings of proteins, according to their context across multiple networks. Using these embeddings, we trained supervised machine learning models to predict protein function in budding and fission yeast. We show that, of the 3×10^4 dimensions in the yeast STRING networks, just 256 dimensions ($< 1\%$) are sufficient to adequately represent each protein. We also found that a vector of protein functions can be predicted using structured learning with the same performance as predicting each function using a separate classifier and the one-vs-rest strategy.

Finally, in Chapter 5, we collected a large, phenomic data set of *Schizosaccharomyces pombe* gene deletion mutant strains grown in 131 different conditions. We trained machine learning models using this bespoke experimental data, alongside orthogonal data from protein network embeddings (from Chapter 4) and evolutionary information from CATH-

FunFams. We evaluated the performance of models trained on these data modes separately, and in combination, finding that the best performing model used a combination of network and evolutionary data. Finally, we entered the predictions from this model into the fourth CAFA protein function prediction competition.

Chapter 2

Dynamic changes in the brain protein interaction network correlates with progression of Aβ42 pathology in *Drosophila*

2.1 Introduction

This chapter is a modified version of the paper: Scholes, H.M. Dynamic changes in the brain protein interaction network correlates with progression of Aβ42 pathology in *Drosophila. Scientific Reports* (2020) [163]. Adam Cryar, Fiona Kerr, David Sutherland, Lee Gethings, Johannes Vissers, Jonathan Lees, Christine Orengo, Linda Partridge and Konstantinos Thalassinos contributed to the research of the original publication. All final language is my own.

The proteomics data set was collected predominantly by Adam Cryar. This project was a collaboration between the Orengo and Thalassinos groups, which consited of me analysing the proteomics data using sophisticated bioinformatics techniques.

2.1.1 Alzheimer's disease

Alzheimer's disease (AD) is a progressive and devastating neurodegenerative disease that is the most prevalent form of dementia [164]. Symptoms initially present as episodic memory loss and subsequently develop into widespread cognitive impairment. Alois Alzheimer first identified his eponymous disease in 1907 during the dissection of a demented brain post-mortem [165]. Two brain lesions are pathological hallmarks of the disease: plaques and neurofibrillary tangles. Plaques are extracellular aggregates of the protein amyloid

beta (Aβ) [166], whereas neurofibrillary tangles are intraneuronal aggregates of hyperphosphorylated tau [167, 168]. In addition to these hallmarks, the AD brain experiences many other changes, including metabolic and oxidative dysregulation [169, 170], DNA damage [171], cell cycle re-entry [172], axon loss [173] and, eventually, neuronal death [170, 174].

Despite a substantial research effort, no cure for AD has been found. Effective treatments are desperately needed to cope with the projected increase in the number of new cases as a result of longer life expectancy and an ageing population. Sporadic onset is the most common form of AD (SAD), for which age is the major risk factor. Familial AD (FAD)—a less common ($< 1\%$), but more aggressive, form of the disease—has an early onset of pathology before the age of 65 [175]. FAD is caused by fully penetrant mutations in the Aβ precursor protein (APP) and two subunits—presenilin 1 and presenilin 2—of the γ-secretase complex that processes APP in the amyloidogenic pathway to produce Aβ. APP is a 770 amino acid integral membrane protein that is involved in a wide range of developmental processes in neurons—–functioning as a cell surface receptor and cell adhesion molecule [176]. Whilst the exact disease mechanisms of AD are not yet fully understood, this has provided support for Aβ accumulation as a key player in its cause and progression [164]. Aβ42—a 42 amino acid variant of the peptide—is neurotoxic [177], necessary for plaque deposition [178] and sufficient for tangle formation [179].

2.1.1.1 Aβ formation

Aβ is formed by cleavage of APP, a 695 to 770 amino acid transmembrane protein expressed in many tissues [180]. In addition to its role in AD, APP performs many cellular functions, notably being a cell surface receptor protein for stimulating intracellular Ser/Thr kinases [181] and as a transcriptional regulator of miRNAs involved in neuronal differentiation [182].

Cleavage of APP can occur via two distinct pathways involving the secretase family of endopeptidases (Fig. 2.1). Most APP is processed in the non-amyloidogenic pathway by the α-secretases that cleave at a residue located 83 amino acids from the C-terminus, near the extracellular membrane surface [183]. Cleavage yields two fragments: a C-terminal fragment CTF83 that remains in the membrane and an sAPPα extracellular protein. Formation of the Aβ peptide is inherently prevented due to the position of the cleavage site in APP. In the complementary amyloidogenic pathway that produces Aβ, APP is cleaved

by the β-secretases at a different position to yield a smaller extracellular protein sAPPβ and a larger membrane protein CTF99 [183]. γ-secretases subsequently cleave CTF99 38 to 43 residues from its newly generated N-terminal residue to liberate the transmembrane region as the Aβ peptide. In addition to APP, FAD mutations are frequently found in components of the γ-secretase complex: presenelin 1 and 2 [183]. Typically the 40 residue long Aβ40 peptide is produced. Approximately 10% of CTF99 is processed to form the 42 residue long Aβ42 peptide [184]. Aβ42 is more hydrophobic than Aβ40 and is more liable to fibril formation [185].

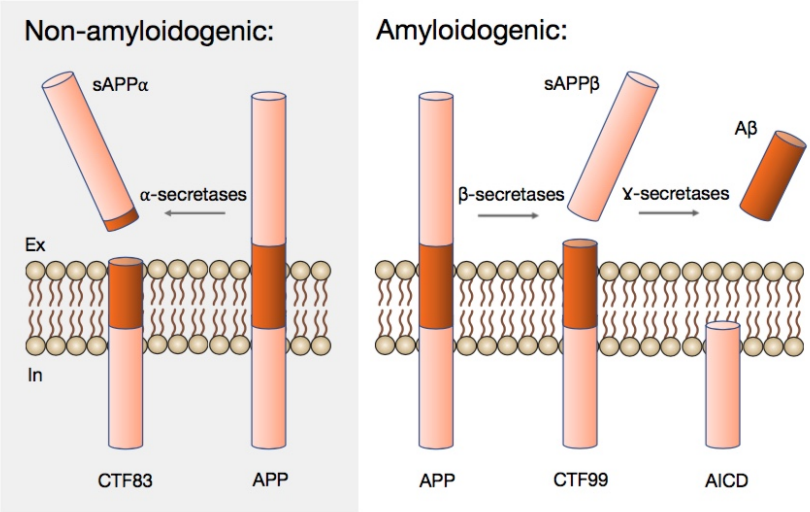


Figure 2.1: **Processing pathways of APP.** Figure adapted from [184].

2.1.1.2 The Arctic mutant

The Arctic mutation in Aβ42 (Glu22Gly) [186] causes a particularly aggressive form of familial AD that is associated with an increased rate and volume of plaque deposition [187]. Genetic analyses of SAD, however, suggest a complex molecular pathology, in which alterations in neuro-inflammation, cholesterol metabolism and synaptic recycling pathways may also be required for Aβ42 to initiate the toxic cascade of events leading to tau pathology and neuronal damage in dementia.

2.1.1.3 Plaque formation

Late-stage AD brains are characterized by insoluble fibrillar deposits of Aβ, known as senile plaques. Plaques are formed in a nucleated event initiated by the pathogenic Aβ42 peptide [185]. Aβ can exist in a number of oligomeric states (Fig. 2.2). Small oligomers act as nucleation centres that can be elongated into protofibrils. Ultimately, protofibrils

aggregate into larger assemblies known as fibrils. Subunits of Aβ polymerize by forming β-sheets perpendicular to the protofibril axis [188]. Monomeric and fibrillar states are innocuous, whilst small oligomers lacking defined secondary structure are neurotoxic [189].

Much debate has surrounded the role of Aβ42 in plaque formation because it constitutes just 10% of the Aβ pool [184], but dominates plaque composition [190]. The first definitive piece of evidence came from a study of mice overexpressing Aβ40 or Aβ42 [178]. Aβ42 was shown to be essential for plaque formation, whilst plaques do not even form with Aβ40. But this is not to say that plaques are homogenous. A proteomics study found that plaques contain on average 913 proteins, of which 279 are consistently found [191].

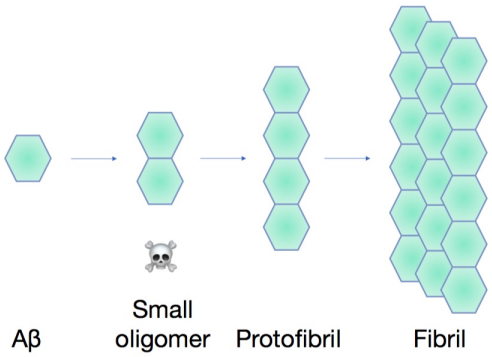


Figure 2.2: **Oligomeric states of Aβ.** Figure adapted from [184].

FAD is linked to mutations that lead to increased accumulation of extracellular Aβ42 [192]. Mutations implicated in FAD have been shown to alter the conformation of Aβ and affect its aggregation [193]. Of particular interest to this study is the Arctic mutant Aβ42 that is associated with decreased levels of Aβ in plasma, an increased rate of aggregation and neurotoxicity [187, 194].

Traditionally, Aβ was thought to only accumulate extracellularly. Intraneuronal amyloid was first identified by immunohistochemistry [195]. More recently, staining with C-terminal specific anti-Aβ antibodies provided evidence of intraneuronal Aβ42 accumulation prior to the formation of plaques and neurofibrillary tangles (NFTs) [196]. Indeed, it is believed that intraneuronal Aβ accumulates first and is the source of Aβ that develops into extracellular plaques [196]. Additionally, intracellular Aβ42 induces cell death by apoptosis, whereas extracellular Aβ42 and intra-/extracellular Aβ40 are nontoxic [177]. Overall it appears that senile plaques are a marker of advanced AD, whereas, smaller oligomers of Aβ42 are pathological.

2.1.1.4 Involvement of tau

tau is a neuronal protein that interacts with tubulin to promote and maintain microtubule assembly [197]. Intraneuronal aggregates of tau form NFTs in the second AD lesion of AD. tau is also implicated in a number of other neurological conditions, known as tauopathies [198]. A link between Aβ and tau has been established, although there is debate as to the importance of these agents in AD progression. Mechanistically, Aβ42 induces the abnormal hyperphosphorylation of tau, which subsequently form paired helical fragment building blocks of NFTs [198]. This was confirmed by injecting Aβ42 into the brains of mice, causing NFTs to form locally [179]. In a similar vain to plaques, whilst NFTs are a hallmark of AD, they appear to be inert [199]. But 40% of hyperphosphorylated tau is monomeric in AD [198] and it is this form that is neurotoxic, sequesters normal tau and promotes the disassembly of microtubules [200]. In summary, soluble Aβ and tau work in conjunction to convert healthy neurons into a diseased state, occurring independently of plaques and NFTs.

2.1.1.5 Proteomics

Post-mortem proteomics studies on human brains have been valuable in adapting the amyloid cascade hypothesis of AD and helped develop the alternative neuro-inflammation hypothesis of AD. These studies revealed that the brain undergoes oxidative damage as a response to amyloid accumulation in the end stages of disease. Using model organisms, such as fruit flies, allows molecular alterations in the brain to be tracked from the onset of AD, during its progression, until death. Comparison of proteomic analyses of post-mortem human brains have further revealed an increase in metabolic processes and reduction in synaptic function in AD [201]. Oxidised proteins also accumulate at early stages in AD brain, probably as a result of mitochondrial ROS production [202], and redox proteomic approaches suggest that enzymes involved in glucose metabolism are oxidised in mild cognitive impairment and AD [203, 204]. Moreover, phospho-proteomic approaches have revealed alterations in phosphorylation of metabolic enzymes and kinases that regulate phosphorylation of chaperones such as HSP27 and crystallin alpha B [205]. Of note, however, there is little proteomic overlap between studies using post-mortem human brain tissue, which may reflect the low sample numbers available for such studies, differences in comorbidities between patients and confounding post-mortem procedures [201]. Although valuable, post-mortem studies also reflect the end-stage of disease and, therefore,

do not facilitate measurement of dynamic alterations in proteins as AD progresses.

2.1.1.6 Animal models

Animal models of AD, generated through transgenic over-expression of human APP or tau, provide an opportunity to track proteomic alterations at pre- and post-pathological stages, thus facilitating insight into the molecular mechanisms underlying disease development and revealing new targets for drugs to prevent AD progression. Analyses of transgenic mice models of AD have revealed some overlapping alterations in metabolic enzymes, kinases and chaperones with human AD brain [201]. Only one study, however, has tracked alterations in protein carbonylation over time, showing increases in oxidation of metabolic enzymes (alpha-enolase, ATP synthase α-chain and pyruvate dehydrogenase E1) and regulatory molecules (14-3-3 and Pin1) in correlation with disease progression [206].

Adult-onset *Drosophila* models of AD have been generated by over-expressing human Aβ42 peptide exclusively in adult fly neurons using inducible expression systems. These models have been shown to develop progressive neurodegenerative phenotypes, such as reduced climbing ability, and shortened lifespan [207]. Taking advantage of the short lifespan of the fly, and the flexible nature of the inducible model, we have performed a longitudinal study of the brain proteome to capture the effects of Aβ42-toxicity in the brain from the point of induction and across life.

2.1.2 Mass spectrometry

Mass spectrometry (MS) is a sensitive analytical technique that was developed in 1912. MS measures the mass-to-charge ratio (m/z) of an analyte. Over the past century, MS has seen rapid development and has created many distinct, but related, fields. This is largely due to MS being able to measure the m/z of analytes whose masses range over several orders of magnitude—from small molecules under 100 Da to a 52 MDa whole virus [208].

There are three main components in any mass spectrometer: the source, analyser and detector. Analyte molecules are ionised to form positive ions by the source. Many different types of source have been developed, including soft and hard ionisation methods, some of which are covered below. All ions with the same charge are then accelerated to the same kinetic energy ($KE = \frac{mv^2}{2}$). By rearranging the equation, we find that the velocity of an ion depends on its mass ($v = \sqrt{\frac{2KE}{m}}$), so heavier ions will have a lower velocity than lighter ions. Analysers exploit this property to sort ions according to m/z. For example, time-of-flight analysers accelerate ions through a flight tube and measure the m/z by the

time it takes for the ion to reach the detector. Finally, any ions that pass through the analyser are detected as an electrical current by the negatively charged detector.

One such burgeoning MS field is proteomics—the study of proteomes by MS [209, 210]. Technological developments in MS that are relevant to proteomics—and particularly the methods used in this study—are introduced below.

2.1.2.1 Soft ionisation

Traditional electron ionisation methods use an electron beam to form positive ions by knocking elctrons off molecules. This method is high-energy and leads to fragmentation of molecules, so is unsuitable for the analysis of proteins. Two soft ionisation methods were developed in the 1980s, which allowed MS to be applied to macromolecules with minimal fragmentation. Matrix-assisted laser desorption/ionisation (MALDI) uses a laser to liberate the analyte from a laser-absorbing matrix [211, 212]. Electrospray ionisation (ESI) applies a high voltage to a liquid containing the analyte, which produces an aerosol [213]. Unlike MALDI, ESI is able to create multiply charged ions, which extends the m/z range that can be detected. In 2002, the inventors of these methods were awarded The Nobel Prize in Chemistry.

2.1.2.2 Ion-mobility

Ion-mobility spectrometry is a method of separating ions in the gas phase [214]. Ions enter a drift tube that is filled with a buffer gas and are moved through the drift tube with an electric field. Particles of the buffer gas collide with the ions, which creates a frictional force that impedes their progress. The time that it takes for a molecule to move through the drift tube is known as the arrival time. The rate of these collisions depends on the ion's collision cross section (CCS), a measure that quantifies the overall shape of a particle. CCS refers to the ensemble of all possible geometric orientations of a particle, and all possible interaction types that it may have with other particles in the gas phase, averaged into a cross sectional area of a circle [215]. Although the CCS of a protein is correlated with its mass, there is a high variance in CCS for any given mass [215]. This makes intuitive sense because proteins are not perfect spheres of uniform density—some are elongated and others have pockets. CCS predicts that spheroidal globular proteins will collide less often with the buffer gas than proteins with a large surface area-to-volume ratio. Furthermore, chemically identical protein molecules do not have exactly the same arrival time, but rather their arrival times are distributed, due to conformational differences. Differential CCS

allows ions to be separated by their arrival time an additional dimension of shape, thus enabling ion-mobility spectrometers to separate ions according to their arrival time, which depends on the m/z and shape of the particle. Abstractly, ion-mobility is somewhat similar to gel filtration chromatography, in that proteins are separated by their shape. Both CCS and arrival time are not inherent properties of particles because they depend on the buffer gas used and the temperature [214].

2.1.2.3 Tandem mass spectrometry

MS-based proteomics typically uses a tandem MS (MS/MS) setup, consisting of MS1 and MS2 stages [216]. Proteins with a particular m/z are selected in MS1 using an analyser, such as an ion-mobility, time-of-flight or quadrupole analyser. The protein is then fragmented in MS2 and the resulting peptides are detected. The combination of the fragment ion's mass spectrum and m/z are sufficient to determine its amino acid sequence [217].

To further improve the performance of MS/MS for proteomics, proteins are first purified by liquid chromatography, so called LC-MS/MS [218]. Liquid chromatography separates proteins by some property, such as size in gel filtration, or polarity in high performance liquid chromatography (HPLC) and ultra performance liquid chromatography (UPLC), due to differences in how proteins in the sample interact with the mobile and stationary phases. Proteins are eluted from chromatography columns and the eluent is directly analysed by the MS instrument. We used a reverse phase UPLC column in this study. Typically HPLC and UPLC use a normal phase column that has a polar stationary phase. Polar molecules will interact more strongly with the stationary phase and will elute slowly, whereas, non-polar molecules will interact weakly and will elute quickly. In reverse phase columns, the stationary phase is non-polar, so the situation is reversed and non-polar molecules interact strongly and elute slowly.

We used a Waters Synapt GS-Si tandem mass spectrometer in this study (Fig. 2.3). This instrument uses a quadrupole, followed by an ion-mobility analyser for MS1 and a time-of-flight analyser for MS2.

2.1.2.4 Quantitative label-free proteomics

The intensity of electric current that is generated by fragment ions in MS2 can be used to estimate the relative abundance of the protein in the sample [216]. So that abundances can be compared across multiple samples, relative abundances are normalised to a reference peptide that is spiked-in at known concentration and converted to meaningful units.

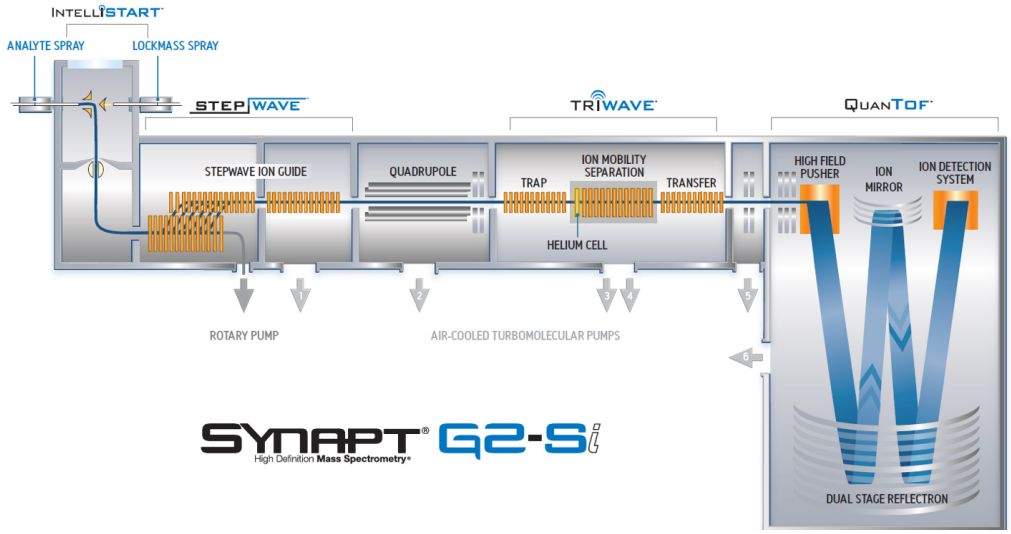


Figure 2.3: **Schematic of the Synapt GS-Si mass spectrometer.** Figure produced by Waters Corporation.

Sample preparations for label-free MS is simpler than labeled methods, but label-based methods remain the most accurate methods for quantification.

2.1.2.5 Data-independent acquisition

Data-independent acquisition (DIA) is a high-throughput and unbiased method of acquiring MS data [219, 220]. Prior to the advent of DIA, data-dependent acquisiton was used, which is low-throughput. In DIA, the entire m/z range is analysed by fragmenting all ions, or predefined m/z windows. If samples are complex, then the peptides that are detected will be highly multiplexed, requiring software analysis tools to deconvolute the data. Modern DIA methods use ion-mobility [221].

2.1.3 Contributions

AD, the most prevalent form of dementia, is a progressive and devastating neurodegenerative condition for which there are no effective treatments. Understanding the molecular pathology of AD during disease progression may identify new ways to reduce neuronal damage. Here, we present a longitudinal study tracking dynamic proteomic alterations in the brains of an inducible *Drosophila melanogaster* model of AD expressing the Arctic mutant Aβ42 gene. We identified 3093 proteins from flies that were induced to express Aβ42 and age-matched healthy controls using label-free quantitative ion-mobility data independent analysis mass spectrometry. Of these, 228 proteins were significantly altered by Aβ42 accumulation and were enriched for AD-associated processes. Network analy-

ses further revealed that these proteins have distinct hub and bottleneck properties in the brain protein interaction network, suggesting that several may have significant effects on brain function. Our unbiased analysis provides useful insights into the key processes governing the progression of amyloid toxicity and forms a basis for further functional analyses in model organisms and translation to mammalian systems.

2.2 Methods

2.2.1 Data collection

2.2.1.1 Fly stocks

The transgenic AD (TgAD) fly line used in this study [207] contains the human transgene encoding the Arctic mutant Aβ42 peptide under the control of an Upstream Activation Sequence (UAS) [222]. Expression of Aβ42 was controlled by GeneSwitch [223]—a mifepristone-inducible GAL4/UAS expression system—under the pan-neuronal elav promoter. All flies were backcrossed for six generations into the w^{1118} genetic background.

Flies were grown in 200 ml bottles on a 12 h/12 h light/dark cycle at constant temperature (25°C) and humidity. Growth media contained 15 g/l agar, 50 g/l sugar, 100 g/l autolysed yeast, 100 g/l nipagin and 3 ml/l propionic acid. Flies were maintained for two days after eclosion before females were transferred to vials at a density of 25 flies per vial for the lifespan analysis and 10 flies per vial for the IM-DIA-MS analysis. Expression of Aβ42 was induced in TgAD flies by spiking the growth media with mifepristone to a final concentration of 200 µM. Flies were transferred to fresh media three times per week, at which point the number of surviving flies was recorded. For each of the three biological repeats, 10 healthy and 10 Aβ42 flies were collected at 5, 19, 31 and 46 days, as well as 54 and 80 days for healthy flies. Following anesthetisation with CO_2, brains were dissected in ice cold 10 mM phosphate buffered saline snap frozen and stored at −80°C.

2.2.1.2 Extraction of brain proteins

Brain proteins were extracted by homogenisation on ice into 50 µl of 50 mM ammonium bicarbonate, 10 mM DTT and 0.25% RapiGest detergent. Proteins were solubilised and disulfide bonds were reduced by heating at 80°C for 20 minutes. Free cysteine thiols were alkylated by adding 20 mM IAA and incubating at room temperature for 20 minutes in darkness. Protein concentration was determined and samples were diluted to a final concentration of 0.1% RapiGest using 50 mM ammonium bicarbonate. Proteins were digested

with trypsin overnight at 37°C at a 50 : 1 protein:trypsin ratio. Additional trypsin was added at a 100 : 1 ratio the following morning and incubated for a further hour. Detergent was removed by incubating at 60°C for one hour in 0.1% formic acid. Insoluble debris was removed by centrifugation at 14,000x g for 30 minutes. Supernatant was collected, lyophilised and stored at −80°C. Prior to lyophilisation peptide concentration was estimated by nanodrop (Thermo Fisher Scientific, Waltham, MA).

2.2.2 Label-free quantitative IM-DIA-MS

Peptides were separated by nanoscale liquid chromatography (LC) by loading 300 ng of protein onto an analytical reversed phase column. IM-DIA-MS analysis was performed using a Synapt G2-Si mass spectrometer (Waters Corporation, Manchester, UK). The time-of-flight analyzer of the instrument was externally calibrated with a NaCsI mixture from m/z 50 to 1,990. Spectra were acquired over a range of 50 to 2,000 m/z. Each biological repeat was analysed at least twice to account for technical variation. LC-MS data were peak detected and aligned by Progenesis QI for proteomics (Waters Corporation). The principles of the embedded search algorithm for DIA data has been described previously [224]. Proteins were identified by searching against the *Drosophila melanogaster* proteome in UniProt, appended with common contaminants, and reversed sequence entries to estimate protein identification false discovery rate (FDR) values, using previously specified search criteria [225]. Peptide intensities were normalised to control for variation in protein loading and relative quantification. Abundances were estimated by Hi3-based quantitation [226].

2.2.2.1 IM-DIA-MS analysis

Nanoscale LC separation of tryptic peptides was performed using a nanoAcquity UPLC system (Waters Corporation) equipped with a UPLC HSS T3 1.7 µm, 75 µm x 250 mm analytical reverse phase column (Waters Corporation). Prior to peptide separation, 300 ng of tryptic peptides were loaded onto a 2G, V/V 5 µm, 180 µm x 20 mm reverse phase trapping column at 5 µl/min for three minutes. IM-DIA-MS analysis of tryptic digests was performed using a Synapt GS-Si mass spectrometer equipped with a T-Wave-IMS device. Mass measurements were made in positive-mode ESI with the instrument operated in resolution mode with a typical resolving power of 20,000 full width at half maximum. Prior to analysis the time-of-flight analyzer was externally calibrated with a NaCsI mixture from m/z 50 to 1,990. The data were post-acquisition lock mass corrected using the double

charged monoisotopic ion of [Glu1]-Fibrinopeptide B. To achieve lock mass correction, a 100 fmol/µl solution of [Glu1]-Fibrinopeptide B was infused at a 90° angle to the analytical sprayer. This reference sprayer was sampled every 60 seconds. Accurate IM-DIA-MS data were collected in the DIA mode of analysis, HDMS^E IM spectrometry was performed by applying a constant wave height of 40 V whilst a constant wave velocity of 650 m/s was maintained. Wave heights within the trap and transfer were both set at 4 V whilst the wave velocities were 311 and 175 m/s respectively. MS data were acquired over 50 to 2,000 m/z for each mode. Spectral acquisition time for each mode was 0.5 s with a 0.015 interscan delay, corresponding to a cycle of low and elevated energy data being acquired every 1.1 s. During the low energy MS mode data was acquired whilst applying a constant collision energy of 4 eV within the transfer. After IMS, MS/MS data was acquired by ramping the collision energy within the transfer region between 15 and 45 eV. To ensure that ions with a m/z less than 350 were derived from peptide fragmentation within the transfer region the radio frequency applied to the quadrupole mass analyser was adjusted to optimise transmission within the region of 350 to 2,000 Da. Each biological replicate was analysed at least twice.

2.2.2.2 MS Data Processing

All MS data were processed in Progenesis QI for proteomics. Data were imported into Progenesis to generate a 3D representation of the data (m/z, retention time and peak intensity). Samples were then time aligned with the software allowed to automatically determine the best reference run from the dataset. Following alignment, peak picking was performed on MS level data. A peak picking sensitivity of 4 (out of 5) was set. Peptide features were tentatively aligned with their respective fragment ions based primarily on the similarity of their chromatographic and mobility profiles. Requirements for features to be included in post-processing database searching were as follows: 300 counts for low energy ions, 50 counts for high energy ions and 750 counts for deconvoluted precursor intensities. Subsequent data were searched against 20,049 sequences from the UniProt canonical *Drosophila* database (appended with common contaminants). Trypsin was specified as the enzyme of choice and a maximum of two missed cleavages were permitted. Carbamidomethyl (C) was set as a fixed modification whilst oxidation (M) and N-terminal acetylation were set as variable modifications. Peptide identifications were grouped and relative quantification was performed using non-conflicting peptides only.

2.2.3 Data analysis

2.2.3.1 Data processing

Proteins that were identified in both healthy and Aβ42 flies were considered for further analysis. Missing data were replaced by the minimum abundance measured for any protein in the same repeat [227]. The data were quantile normalised [228], so that different conditions and time points could be compared reliably. Quantile normalisation transforms the abundances so that each repeat has the same distribution.

For principal component analysis (PCA) analysis, the data were $\log_{10}$-transformed and each protein was standardised to zero mean and unit variance. Hierarchical biclustering was performed using the Euclidean distance metric with the complete linkage method. Prior to clustering, proteins were normalised to their abundance in healthy flies at 5 days.

Proteins that were identified by IM-DIA-MS in either healthy or Aβ42 flies were assessed for overrepresentation of Gene Ontology (GO) terms using GOrilla [228], which uses ranked lists of target and background genes. Proteins were ranked in descending order by their mean abundance. The type I error rate was controlled by correcting for multiple testing using the Benjamini-Hochberg method at an FDR of 5%. Clusters of proteins were assessed for overrepresentation of GO-Slim terms in the Biological Process ontology using Panther (version 13.1) with a custom background of the $3{,}093$ proteins identified by IM-DIA-MS in healthy or AD flies.

2.2.3.2 Identification of significantly altered proteins

Significantly altered proteins were identified using five methods that are frequently used to identify differentially expressed genes in time course RNA-Seq data. DESeq2 [229], EDGE [230], edgeR [231], limma [232] and maSigPro [233] are all available in R through Bioconductor. Dispersions were estimated from the biological and technical repeats. Unless otherwise stated, default parameters were used for all methods under the null hypothesis that a protein does not change in abundance between healthy and AD conditions in normal ageing. The type I error rate was controlled by correcting for multiple testing using the Benjamini-Hochberg method at a FDR of 5%. A protein was classified as significantly altered if two or more methods identified it.

DESeq2 models proteins with the negative binomial distribution and performs likelihood ratio tests. A time course experiment was selected in EDGE using the likelihood ratio test and a normal null distribution. edgeR uses the negative binomial distribution

and performs quasi-likelihood tests. limma fits linear models to the proteins and performed empirical Bayes F-tests. maSigPro fits generalised linear models to the proteins and performs log-likelihood ratio tests.

Significantly altered proteins were clustered using a Gaussian mixture model. Protein abundances were $\log_{10}$-transformed and z scores were calculated. Gaussian mixture models were implemented for 1 to 228 clusters. The best model was chosen using the Bayesian information criterion (BIC), which penalises complex models BIC $= -2\ln(L) + \ln(n)k$, where $\ln(L)$ is the log-likelihood of the model, n is the number of significantly altered proteins and k is the number of clusters. The model with lowest BIC was chosen.

2.2.3.3 Networks

All network analysis was performed using the *Drosophila melanogaster* Search Tool for the Retrieval of Interacting Genes/Proteins (STRING) network (version 10) [234]. Low confidence interactions with a 'combined score' < 0.5 were removed in all network analyses.

Network properties of the significantly altered proteins were analysed in the brain protein interaction network. A subgraph of the STRING network was induced on the 3,093 proteins identified by IM-DIA-MS in healthy or Aβ42 flies and the largest connected component was selected (2,428 nodes and 44,561 edges). The subgraph contained 183 of the 228 significantly altered proteins. For these proteins, four network properties were calculated as test statistics: mean node degree; mean unweighted shortest path length between a node and the remaining 182 nodes; the size of the largest connected component in the subgraph induced on these nodes; and mean betweenness centrality. Hypothesis testing was performed using the null hypothesis that there is no difference between the nodes in the subgraph. Assuming the null hypothesis is true, null distributions of each test statistic were simulated by randomly sampling 183 nodes from the network 10,000 times. Using the null distributions, one-sided non-parametric P values were calculated as the probability of observing a test statistic as extreme as the test statistic for the significantly altered proteins.

A subgraph of the STRING network was induced on the proteins significantly altered in AD and their neighbours and the largest connected component was selected (4,842 nodes and 182,474 edges). The subgraph contained 198 of the 228 significantly altered proteins and was assessed for enrichment of GO terms. Densely connected subgraphs were identified using MCODE [235]. Modules were selected with an MCODE score $>$

10. As STRING is a functional interaction network, clusters of nodes may correspond to proteins from the same complex, pathway or functional family. Clusters were assessed for overrepresentation of GO Slim terms in the Biological Process ontology using Panther [236] (version 13.1) with a custom background of the 3,093 proteins identified by IM-DIA-MS in healthy or Aβ42 flies. Fisher's exact tests were performed and the type I error rate was controlled by correcting for multiple testing using the Benjamini-Hochberg method at an FDR of 5%.

2.2.3.4 Open source software

Data analysis was performed in Python v3.6 (Python Software Foundation, http://www.-python.org) using SciPy [237], NumPy [238], Pandas [239], scikit-learn [240], NetworkX [241], IPython [242] and Jupyter [243]. Figures were plotted using Matplotlib [244] and seaborn.

2.3 Results

2.3.1 Proteome analysis of healthy and Aβ42-expressing fly brains

In this study, we used an inducible transgenic fly line expressing human Arctic mutant Aβ42 (TgAD) [222, 223] (Fig. 2.4a). Flies were either chronically induced two days after eclosion (Aβ42 flies) or remained uninduced (healthy flies) and used as a control of normal ageing.

We first sought to determine how the fly lifespan is affected in TgAD. We confirmed a previously observed [207] reduction in lifespan following Aβ42 induction prior to proteomic analyses (Fig. 2.4b).

To understand how the brain proteome is affected as Aβ42 toxicity progresses, fly brains were dissected from healthy and Aβ42 flies at 5, 19, 31 and 46 days, and at 54 and 80 days for healthy controls, then analysed by label-free quantitative IM-DIA-MS (Fig. 2.4c). 1,854 proteins were identified in both healthy and Aβ42 fly brain from a total of 3,093 proteins (Fig. 2.4d), which is typical for recent fly proteomics studies [245, 246].

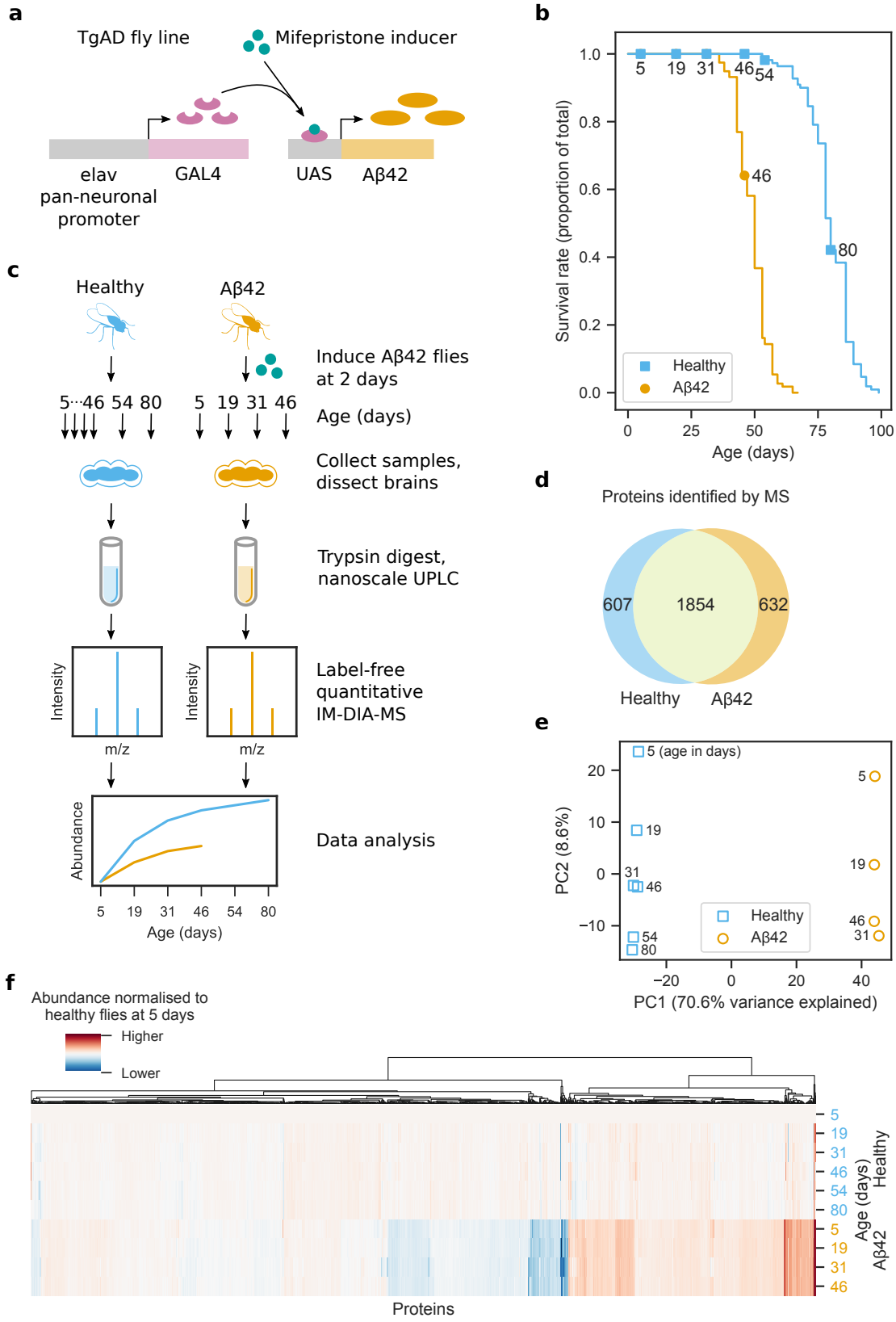


Figure 2.4: **Proteome analysis of healthy and AD fly brains.** *Cont.*

Figure 2.4: **Proteome analysis of healthy and AD fly brains.** (a) *Drosophila melanogaster* transgenic model of AD (TgAD) that expresses Arctic mutant Aβ42 in a mifepristone-inducible GAL4/UAS expression system under the pan-neuronal elav promoter. (b) Survival curves for healthy and Aβ42 flies. Aβ42 flies were induced to express Aβ42 at 2 days. Markers indicate days that MS samples were collected. (c) Experimental design of the brain proteome analysis. Aβ42 flies were induced to express Aβ42 at 2 days. For each of the three biological repeats, 10 healthy and 10 Aβ42 flies were collected at 5, 19, 31 and 46 days, and at 54 and 80 days for healthy controls. Proteins were extracted from dissected brains and digested with trypsin. The resulting peptides were separated by nanoscale liquid chromatography and analysed by label-free quantitative IM-DIA-MS. (d) Proteins identified by IM-DIA-MS. (e) Principal component analysis of the IM-DIA-MS data. Axes are annotated with the percentage of variance explained by each principal component. (f) Hierarchical biclustering using relative protein abundances normalised to their abundance in healthy flies at 5 days.

For the 1,854 proteins identified in both healthy and Aβ42 flies, we assessed the reliability of our data. Proteins were highly correlated between technical and biological repeats We used principal component analysis of the protein abundances to identify sources of variance (Fig. 2.4e). Healthy and Aβ42 samples are clearly separated in the first principal component, probably due to the effects of Aβ42. In the second principal component, samples are separated by increasing age, due to age-dependent or disease progression changes in the proteome. These results show that whilst ageing does contribute to changes in the brain proteome (8.7% of the total variance), much larger changes are due to expression of Aβ42 (70.6%) and this may reflect either a correlation with the ageing process or progression of AD pathology. We confirmed this result using hierarchical biclustering of protein abundances in Aβ42 versus healthy flies at 5 days (Fig. 2.4f). The results reveal that most proteins do not vary significantly in abundance with age in healthy flies, but many proteins are differentially abundant in Aβ42 flies.

2.3.2 Analysis of brain proteome dysregulation in Aβ42 flies

We next identified the proteins that were significantly altered following Aβ42 expression in the fly brain. To achieve this, we used five methods commonly used to analyse time course RNA-Seq data [247] and classified proteins as significantly altered if at least two methods detected them [248]. We identified 228 significantly altered proteins from 740 proteins that were detected by one or more methods (Fig. 2.5a). A comparison of popular RNA-Seq analysis tools [249] showed that edgeR [231] has a high false positive rate and variable performance on different data sets, whereas, DESeq2 [229] and limma [232] have

low false positive rates and perform more consistently. We observed a similar trend in our data set. limma and DESeq2 detected the lowest number of proteins, with 21 proteins in common (Fig. 2.6a). edgeR detected more proteins, of which 38 were also detected by DESeq2 and 16 by limma. EDGE [230] and maSigPro [233] detected vastly more proteins, 464 of which were only detected by one method. Principal component analysis shows that edgeR, DESeq2 and limma detect similar proteins, whereas, EDGE and maSigPro detect very different proteins (Fig. 2.6b).

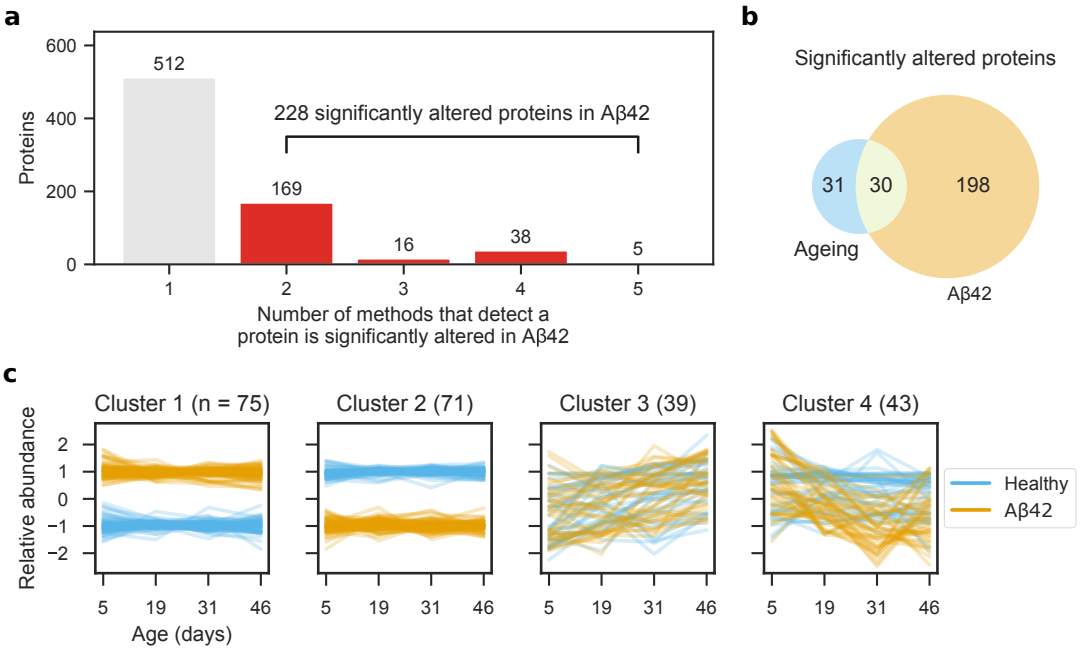


Figure 2.5: **Brain proteome dysregulation in AD.** (a) Proteins significantly altered in AD were identified using five methods (EDGE, edgeR, DESeq2, limma and maSigPro) and classified as significantly altered if at least two methods detected them. (b) Significantly altered proteins in AD (from a) and ageing. (c) Significantly altered protein abundances were z score-transformed and clustered using a Gaussian mixture model.

Although these methods should be able to differentiate between proteins that are altered in Aβ42 flies from those that change during normal ageing, we confirmed this by analysing healthy flies separately. In total, 61 proteins were identified as significantly altered with age, of which 30 were also identified as significantly altered in AD (Fig. 2.5b) and 31 in normal ageing alone. These proteins are not significantly enriched for any pathways or functions. Based on our results, we concluded that the vast majority of proteins that are significantly altered in AD are not altered in normal ageing and that AD causes significant dysregulation of the brain proteome.

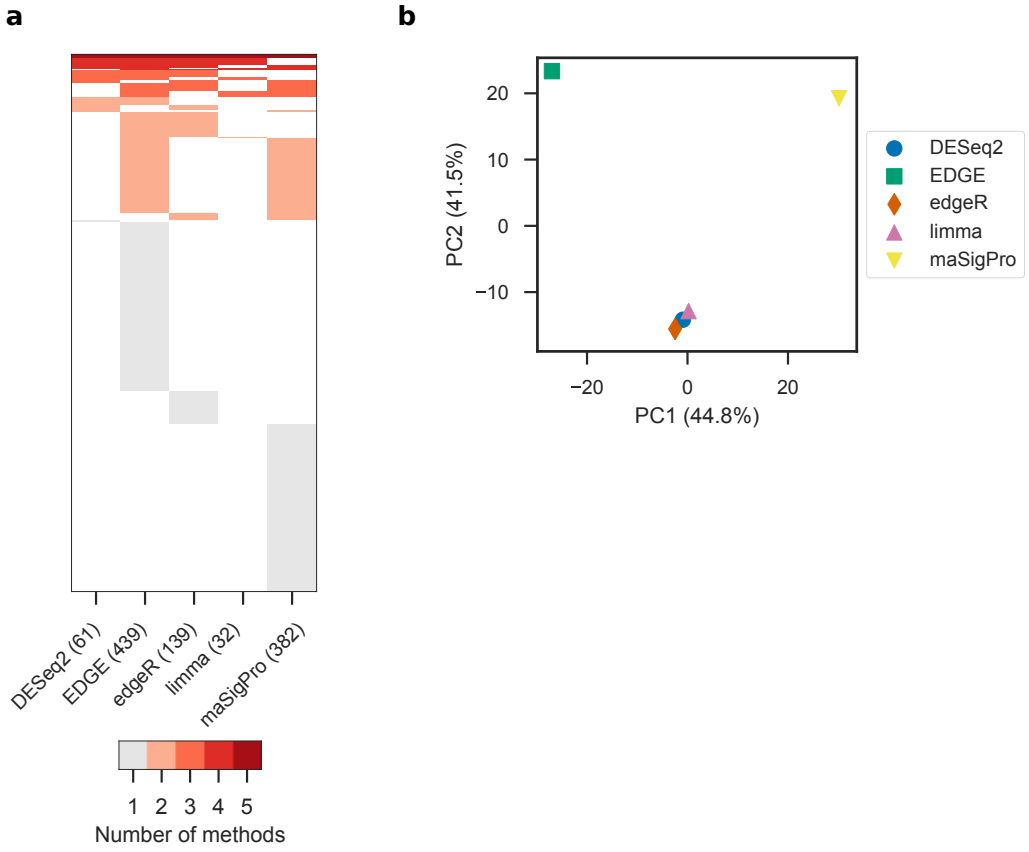


Figure 2.6: **Analysis of the five statistical methods used to identify significantly altered proteins.** (a) Heat map of the proteins detected by each method. (b) Principal component analysis of these results. Axes are annotated with the percentage of variance explained by each principal component.

Of the 31 proteins specifically altered in ageing, 10 decreased with age (Acp1, CG7203, mRPL12, qm, CG11017, HIP, HIP-R, PP02 and Rpn1), 15 increased with age (ade5, CG7352, RhoGAP68F, CG9112, PCB, Aldh, D2hgdh, CG7470, CG7920, RhoGDI, Aldh7A1, CG8036, Ssadh, muc and FKBP14) and four fluctuated throughout life (CG14095, His2A, RpL6 and SERCA).

To understand the dynamics of protein alterations following Aβ42 induction, we clustered the profiles of proteins significantly altered in Aβ42 flies using a Gaussian mixture model (Fig. 2.5c). The proteins clustered best into four sets, determined by the BIC, which was smallest with four clusters. In comparison to healthy flies, cluster 1 contains proteins that have consistently higher abundance in Aβ42 flies. Conversely, cluster 2 contains proteins that have lower abundance in Aβ42 flies. The abundances of proteins from clusters 1 and 2 are affected from the onset of disease at day 5, and remain at similar levels as the

disease progresses. Proteins in cluster 3 follow a similar trend in healthy and Aβ42 flies and increase in abundance with age. However, cluster 4 proteins decrease in abundance as the disease progresses, whilst remaining steady in healthy flies.

We performed a statistical GO enrichment analysis on each cluster, but found no enrichment of terms. Furthermore, we also saw no enrichment when we analysed all 228 proteins together.

2.3.3 Brain proteins significantly altered by Aβ42 have distinct network properties

Following the analyses of brain proteome dysregulation in Aβ42 flies, we analysed the 228 significantly altered proteins in the context of the brain protein interaction network to determine whether their network properties are significantly different to the other brain proteins. We used a subgraph of the STRING [234] network induced on the 3,093 proteins identified by IM-DIA-MS (see Section 2.2.3.3 for more details). This subgraph contained 183 of the 228 significantly altered proteins. We then calculated four graph theoretic network properties (Fig. 2.7a) of these 183 significantly altered proteins contained in this network:

- degree, the number of edges that a node has,
- shortest path, the smallest node set that connect any two nodes
- largest connected component, the largest node set for which all nodes have at least one edge to any of the other nodes;
- betweenness centrality, the proportion of all shortest paths in the network that a particular node lies on

We performed hypothesis tests and found that these proteins have statistically significant network properties. Firstly, the significantly altered proteins make more interactions than expected (mean degree $P < 0.05$; Fig. 2.7b). Therefore, these proteins may further imbalance the proteome by disrupting the expression or activity of proteins they interact with. Secondly, not only are these proteins close to each other (mean shortest path $P < 0.05$; Fig. 2.7c), but also 129 of them form a connected component (size of largest connected component $P < 0.01$; Fig. 2.7d). These two pieces of evidence suggest that Aβ42 disrupts proteins at the centre of the proteome. Lastly, these proteins lie along shortest paths between many pairs of nodes (mean betweenness centrality $P < 0.01$; Fig. 2.7e) and may control how signals are transmitted in cells. Proteins with high betweenness cen-

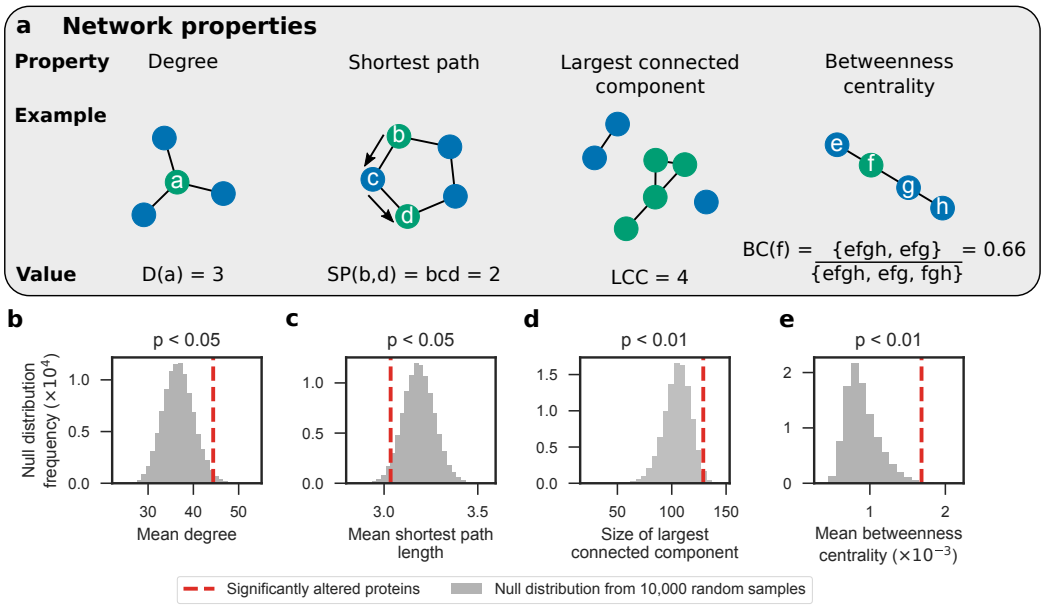


Figure 2.7: **Significantly altered proteins have statistically significant network properties in the brain protein interaction network.** (a) Network properties that were calculated: degree, the number of edges that a node has; shortest path, the smallest node set that connect any two nodes; largest connected component, the largest node set for which all nodes have at least one edge to any of the other nodes; and betweenness centrality, the proportion of all shortest paths in the network that a particular node lies on. Using a subgraph of the STRING network induced on the 3,093 proteins identified by IM-DIA-MS in healthy and Aβ42 flies, the significance of four network characteristics were calculated for the 183 significantly altered proteins contained in this subgraph. (b) mean degree; (c) mean shortest path length between a node and the remaining 182 nodes; (d) the size of the largest connected component in the subgraph induced on these nodes; and (e) mean betweenness centrality. One-sided non-parametric P values were calculated using null distributions of the test statistics, simulated by randomly sampling 183 nodes from the network 10,000 times.

trality are also more likely to be essential genes for viability [250]. Taken together, these findings suggest that the proteins significantly altered in AD are important in the protein interaction network.

2.3.4 Predicting the severity of Aβ42-induced protein alterations using network properties

We predicted how severely particular Aβ42-associated protein alterations may affect the brain using two network properties—the tendency of a node to be a hub or a bottleneck. In networks, nodes with high degree are hubs for communication, whereas nodes with high betweenness centrality are bottlenecks that regulate how signals propagate through the network. Protein expression tends to be highly correlated to that of its neighbours in the protein interaction network. One exception to this rule, however, are bottleneck

proteins, whose expression tends to be poorly correlated with that of its neighbours [250]. This suggests that the proteome is finely balanced and that the expression of bottleneck proteins is tightly regulated to maintain homeostasis. We analysed the hub and bottleneck properties of the significantly altered proteins and identified four hub-bottlenecks and five nonhub-bottlenecks that correlate with Aβ42 expression (Fig. 2.8a) and analysed how their abundances change during normal ageing and as pathology progresses (Fig. 2.8b).

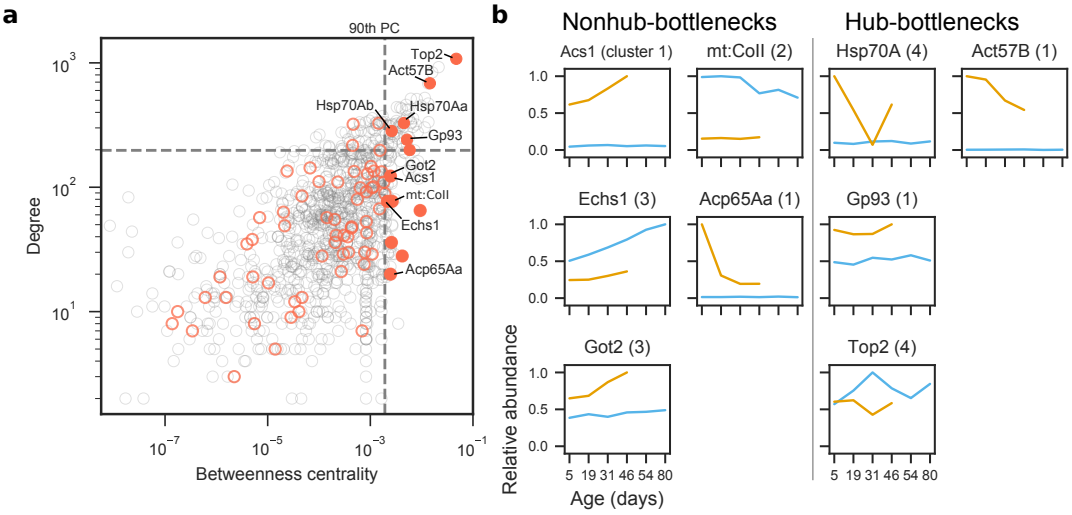


Figure 2.8: **Analysis of hubs and bottlenecks in the brain protein interaction network.** In networks, nodes with high degree are hubs and nodes with high betweenness centrality are bottlenecks. (a) Degree (hub-ness) is plotted against betweenness centrality (bottleneck-ness) in the brain protein interaction network for all proteins identified by IM-DIA-MS (grey circles). Of the significantly altered proteins (red circles), hub-bottleneck ($> 90^{th}$ percentile (PC) for degree and betweenness centrality) and nonhub-bottleneck proteins ($> 90^{th}$ PC for betweenness centrality) are highlighted (filled red circles). (b) Profiles of significantly altered bottleneck proteins implicated in Aβ42 toxicity. Maximum abundances are scaled to unity. Numbers in parentheses denote which cluster from Fig. 2.5c the protein was in.

Non-hub bottleneck proteins (Fig. 2.8b) Acs1 and Got2 levels were stably expressed throughout normal ageing in our healthy flies but increased upon Aβ42 induction and continued to rise with age in Aβ42 flies. On the other hand, Echs1 abundance increased in healthy flies during normal ageing, but its levels were reduced upon Aβ42 induction and its ageing-dependent increase was diminished in Aβ42 flies compared to controls. Levels of mt:CoII (a COX subunit) declined with age in healthy control, but not in Aβ42, fly brain, although its expression was downregulated compared to controls at all time-points following Aβ42 induction. Finally, the cuticle protein Acp65Aa was also upregulated in Aβ42 flies compared to controls, but levels fell sharply between 5 and 19 days of age.

Of the four hub-bottlenecks (Fig. 2.8b) Hsp70A was significantly upregulated at early time-points (5 days) in Aβ42 flies, dropped between days 5 and 31 post-induction, then increased at later time-points, compared to healthy controls which exhibited stable expression of this protein throughout life. We found that Gp93 was increased across age in Aβ42 flies compared to controls, possibly suggesting an early and sustained protective mechanism against Aβ42-induced damage. DNA topoisomerase 2 (Top2), an essential enzyme for DNA double-strand break repair, was decreased in Aβ42 flies, following a pattern which mirrors changes in its expression with normal ageing. Finally, we found that actin (Act57B) was increased in Aβ42 flies but declined with age, in comparison to control fly brains which displayed stable expression across life.

Due to the importance of these hub and bottleneck proteins in the protein interaction network, we predict that AD-associated alterations in their abundance will likely have a significant effect on the cellular dynamics of the brain.

2.3.5 Dysregulated proteins are associated with known AD and ageing network modules

Finally, we clustered the protein interaction network into modules and performed a GO enrichment analysis on modules that contained any of the 228 significantly altered proteins. We saw no GO term enrichment when we tested these proteins clustered according to their abundance profiles (Fig. 2.5c), presumably because the proteins affected in AD are diverse and involved in many different biological processes. However, by testing network modules for functional enrichment, we exploited the principle that interacting proteins are functionally associated. Using a subgraph of the STRING network containing the significantly altered proteins and their directly-interacting neighbours (Section 2.2.3.3), we used MCODE [235] to find modules of densely interconnected nodes. We chose to include neighbouring proteins to compensate for proteins that may not have been detected in the MS experiments due to the stochastic nature of observing peptides and the wide dynamic range of biological samples [227]. The resulting subgraph contained 4,842 proteins, including 183 of the 228 significantly altered proteins, as well as 477 proteins that were only identified in healthy or Aβ42 flies and 3,125 proteins that were not identified in our IM-DIA-MS experiments.

12 modules were present in the network (Fig. 2.9a). Module sizes range from 302 nodes to 17 nodes. The proportion of these modules that were composed of significantly

altered proteins ranged from 0 to 8%. All but one of the modules were enriched for processes implicated in AD and ageing (Fig. 2.9), including respiration and oxidative phosphorylation, transcription and translation, proteolysis, DNA replication and repair, and cell cycle regulation. These modules contained two proteins that were recently found to be significantly altered in the brain of AD mice [251] and are both upregulated four-fold in AD: adenylate kinase, an adenine nucleotide phosphotransferase, and the armadillo protein Arm, involved in creating long-term memories. ApoB was found in the second highest scoring module that contains proteins involved in translation and glucose transport [252] (Fig. 2.9).

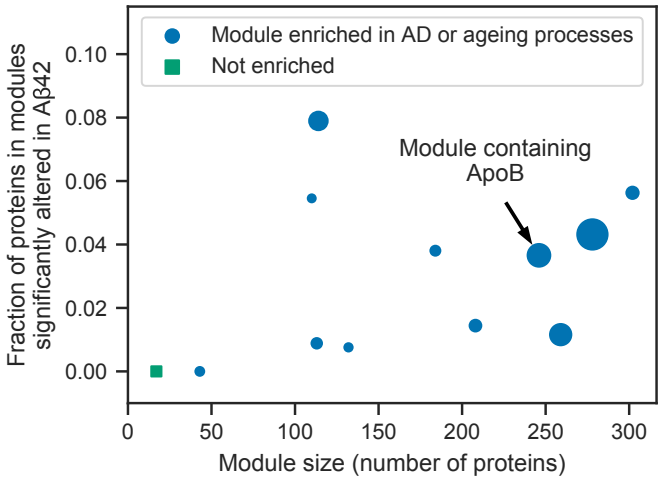


Figure 2.9: **Analysis of network modules enriched for AD or ageing processes.** MCODE was used to identify network modules in a subgraph of the STRING network containing the significantly altered proteins and their directly-interacting neighbours. The size of the resulting 12 modules is plotted against the fraction of proteins in these modules that are significantly altered in AD. Module 2 is annotated as containing ApoB. Marker sizes denote the MCODE score for the module.

In humans, the greatest genetic risk factor for AD is the $\epsilon 4$ allele of ApoE—an apolipoprotein involved in cholesterol transport and repairing brain injuries [253]. A recent study showed that ApoE is only upregulated in regions of the mouse brain that have increased levels of Aβ [251], indicating a direct link between the two proteins. Although flies lack a homolog of ApoE, they do possess a homolog of the related apolipoprotein ApoB (Apolpp) [254], which contributes to AD in mice [255, 256] and is correlated with AD in humans [257, 258]. Interestingly, whilst it was not identified by IM-DIA-MS, ApoB interacts with 12 significantly altered proteins in the STRING network, so is included in

the subgraph induced on the significantly altered proteins and their neighbours. ApoB was found in the second highest scoring module that contains proteins involved in translation and glucose transport [252] (Fig. 2.9).

We analysed the 31 proteins significantly altered in normal ageing, but not AD. Of the 29 proteins that were contained in the STRING network, 24 interact directly with at least one of the AD significantly altered proteins, suggesting an interplay between ageing and AD at the pathway level. Using a subgraph of the STRING network induced on these proteins and their 1,603 neighbours, we identified eight network modules that were enriched for ageing processes [259], including respiration, unfolded protein and oxidative damage stress responses, cell cycle regulation, DNA damage repair, and apoptosis.

2.4 Discussion

Despite the substantial research effort spent on finding drugs against AD, effective treatments remain elusive. We need to better understand the molecular processes that govern the onset and progression of the complex pathologies observed in this condition to AD. This knowledge will help to identify new drug targets to treat and prevent AD.

2.4.1 AD models capture the temporal effects of AD

Analysis of post-mortem human brain tissue is an important way to study dementia, but cannot capture the progression of pathology from the initiation of disease. Due to their short lifespan and ease of genetic manipulation, model organisms such as *Drosophila melanogaster* provide a tractable system in which to examine the progression of AD pathology across life. We performed a longitudinal study of the *Drosophila* brain proteome, using an inducible model of AD, label-free quantitative IM-DIA-MS and network analyses. We were able to track alterations in protein levels from the point of exposure to human Aβ42 and the widespread interaction of Aβ42 with brain signalling networks as pathology progresses.

2.4.2 The brain proteome becomes dysregulated with age, in the absence of AD

We identified 61 proteins which were significantly altered with age in fly brain, 31 of which were not altered in response to Aβ42. Of these, structural chitin proteins (Acp1 and CG7203), mitochondrial associated proteins (HemK1 and mRPL12), geranylgeranyl transferases (qm), and proteostasis proteins (HIP, HIP-R and Rpn1) were significantly downreg-

ulated with age. Indeed, loss of mitochondrial function and proteostasis are key features of the ageing brain [260] in agreement with these observations. Recent studies also suggest a role for geranylgeranyl transferase I-mediated protein prenylation in mediating synaptogenesis and learning and memory [261, 262]. Our findings suggest that this may represent a novel mechanism of regulation and maintenance of these functions during ageing, which warrants further investigation.

Proteins involved in inositol monophosphate (IMP) biosynthesis (ade5), endosome recycling (RhoGAP68F and RhoGDI), oxidation-reduction processes (D2hgdh, Aldh and Aldh7A1), pyruvate metabolism (PCB and muc), neurotransmitter function (Ssadh) and ER-related protein folding (FKBP14) were increased with age in fly brain in our study. Oxidative damage is a key feature of ageing brain and loss of aldehyde dehydrogenase (Aldh) function, which detoxifies oxidative stress inducing aldehydes, has been shown to be associated with promoting age-related neurodegeneration and cognitive dysfunction in mice [263, 264]. Phosphatidylinositol signalling is important for stabilising mood and behaviour, and IMP inhibition is thought to partially mediate the beneficial effects of lithium in the treatment of bipolar disorder [265].

Rho GTPases are involved in maintenance of synaptic function [261], and reductions in their levels correlate with ageing and increases in their expression with foraging behaviour in the brain of honey bees [266]. Our finding that inhibitors of these enzymes (Rho GAPs and Rho GDIs) are upregulated in ageing fly brain further suggests that changes in their activity may mediate loss of synaptic function throughout life.

Ageing is also characterised by a progressive loss of metabolic function, and studies suggest that ATP-generating metabolites, such as pyruvate, improve cognitive function [267] as reflected by upregulation of pyruvate carboxylase and muc, a component of the pyruvate dehydrogenase complex, in our data set.

Finally, several proteins fluctuated in expression across age in our flies, including those involved in DNA repair (His2A), protein translation (RpL6), and ER calcium homeostasis (SERCA), processes which have been previously reported in association with brain ageing [268–270]. Although alterations in these proteins are independent of Aβ42 expression in our flies, further work is required to investigate their functional role in preserving brain function with age and their potential to increase the vulnerability of the ageing brain to neurodegenerative diseases.

2.4.3 AD is associated with widespread dysregulation of the brain proteome

Our proteomic analyses identified Aβ42-induced alterations in levels of 228 proteins, which clustered into four groups. First, those which were either elevated (cluster 1) or reduced (cluster 2) in AD relative to controls throughout life, and dysregulation of which may initiate AD pathogenesis, be involved in early stages of disease progression, or represent defense mechanisms that could be harnessed for protection. Second, those which were altered in correlation with ageing in healthy and Aβ42 flies (cluster 3). Finally, those which changed in Aβ42 flies across life but independently of ageing-dependent effects in healthy controls (cluster 4). Further work is required to determine whether reduction of these proteins plays a causal role in disease pathogenesis that could be targeted therapeutically, or whether their decline represents a protective response to damage.

2.4.4 Brain proteins significantly altered by Aβ42 have distinct network properties

Moreover, computational analysis of these proteins revealed significant network properties within the fly brain proteome. Assessing hub and bottleneck properties, many of the Aβ42-induced proteomic changes represented alterations in bottleneck proteins suggesting that they play key roles in downstream cellular function. Of these, some display non-hub properties indicating that they are important for maintaining cellular homeostasis in a targeted fashion, whereas others also displayed hub properties, suggesting that they are central in linking cellular signalling pathways to maintain cell function.

2.4.5 Proteins dysregulated in AD are hubs and bottlenecks in protein networks

We identified five nonhub-bottleneck proteins and four hub-bottleneck proteins, the expression of which was altered in Aβ42 flies relative to controls across life. Due to the importance of these hub and bottleneck proteins in the protein interaction network, we predict that AD-associated alterations in their abundance will likely have a significant effect on the cellular dynamics of the brain. Indeed, some of these proteins play key molecular roles in metabolism (AscI, Echs1, Got2), protein homeostasis (Hsp70A, Gp93), protection against oxidative stress (mt:CoII), and DNA damage (Top2). These processes have been shown to affect neuronal function and protection against proteo-toxicity.

Acyl-CoA synthetase long chain (Acs1), Enoyl-CoA hydratase, short chain 1 (Echs1),

and Aspartate aminotransferase (Got2), are metabolic enzymes with previous links to neuronal function and damage [169, 170]. Got2 produces the neurotransmitter L-glutamate from aspartate, is involved in assembly of synapses and becomes elevated following brain injury [271].

Cytochrome c oxidase (COX), complex IV of the mitochondrial electron transport chain, uses energy from reducing molecular oxygen to water to generate a proton gradient across the inner mitochondrial membrane. Although Aβ is known to inhibit COX activity [272], the link between COX and AD is unclear. In AD patients, COX activity—but not abundance—is reduced, resulting in increased levels of ROS [273]. However, in COX-deficient mouse models of AD, plaque deposition and oxidative damage are reduced [274].

Hsp70A is a heat shock protein that responds to hypoxia and Gp93 is a stress response protein that binds unfolded proteins, consistent with responses to abnormal Aβ42 aggregation in our flies.

DNA topoisomerase 2 (Top2) is an essential enzyme for DNA double-strand break repair. Double-strand breaks occur naturally in the brain as a consequence of neuronal activity—an effect that is aggravated by Aβ [171]. As a consequence of deficient DNA repair machinery, deleterious genetic lesions may accumulate in the brain and exacerbate neuronal loss.

The cuticle protein Acp65Aa is a chitin, this class of which have been detected in AD brains and suggested to facilitate Aβ nucleation [275].

Finally, actin (Act57B) is a structural protein, and depolymerisation of F-actin filaments in a mouse AD model before onset of AD pathology [276]. Alterations in these proteins may represent either adaptive responses to the presence of abnormal protein aggregates, such as Aβ42, or mediators of neuronal toxicity. Further functional genomic studies are therefore required to establish the causal role of these processes in governing onset and progression of AD pathology.

Assessing the human orthologs of these genes, identified using DIOPT [277], indicates that several of these bottleneck proteins have been previously implicated in association with AD or other neurological conditions in humans or mammalian models of disease. ACSL4 (Acs1 ortholog) has been shown to associate with synaptic growth cone development and mental retardation [278]. Mutations in ECHS1 (Echs1 ortholog), an en-

zyme involved in mitochondrial fatty acid oxidation, associate with Leigh Syndrome, a severe developmental neurological disorder [279]. Proteomic studies have revealed that GOT2 (Got2 ortholog) is down-regulated in infarct regions following stroke [280], and in AD patient brain [281]. Integrating data from human post-mortem brain studies, HSPA1A (Hsp70Aa ortholog) upregulates in the protein interaction network of AD patients compared to healthy controls [282], and has recently been suggested to block APP processing and Aβ production in mouse brain [283]. Synthetic, fibrillar, Aβ42 reduces expression of TOP2B (Top2 ortholog) in rat cerebellar granule cells and in a human mesenchymal cell line, suggesting this may contribute to DNA damage in response to amyloid [284]. HSP90B1 (Gp93 ortholog) shows increased expression following TBI in mice [285], and associates with animal models of Huntington's disease [286]. Finally, ACTB (Act57B ortholog) has been implicated as a significant AD risk gene and central hub node using integrated network analyses across GWAS [287].

2.4.5.1 Nonhub-bottlenecks: Acs1, Echs1, Got2, mt:CoII and Acp65Aa

Three of the nonhub-bottlenecks, Acyl-CoA synthetase long chain (Acs1), Enoyl-CoA hydratase, short chain 1 (Echs1), and Aspartate aminotransferase (Got2), are metabolic enzymes with previous links to neuronal function and damage. Acs1 and Echs1 are involved in the production of acetyl-CoA from fatty acids. Many enzymes involved in acetyl-CoA metabolism associate with AD leading to acetyl-CoA deficits in the brain and loss of cholinergic neurons [170]. Got2 produces the neurotransmitter L-glutamate from aspartate, is involved in assembly of synapses and becomes elevated following brain injury [271]. Brain Acs1 and Got2 levels were stably expressed throughout normal ageing in our healthy flies but increased upon Aβ42 induction and continued to rise with age in Aβ42 flies. This suggests that levels of these proteins increase independently of ageing in AD, but correlate closely with disease progression. On the other hand, Echs1 abundance increases in healthy flies during normal ageing, but its levels were reduced upon Aβ42 induction and its ageing-dependent increase was diminished in Aβ42 flies compared to controls. This may reflect a protective response with ageing that is suppressed by Aβ42 toxicity.

Cytochrome c oxidase (COX), complex IV of the mitochondrial electron transport chain, uses energy from reducing molecular oxygen to water to generate a proton gradient across the inner mitochondrial membrane. Levels of mt:CoII (a COX subunit) declined in aged healthy control fly brain. mt:CoII expression was downregulated in Aβ42 flies

compared to controls at all time-points and was stably-expressed across age following Aβ42 induction. The link between COX and AD is unclear, although Aβ is known to inhibit COX activity [272]. For example, in AD patients, COX activity—but not abundance—is reduced, resulting in increased levels of ROS [273]. However, in COX-deficient mouse models of AD, plaque deposition and oxidative damage are reduced [274]. Hence, the ageing-dependent decline in mt:CoII may represent either a reduction in COX function which renders the brain vulnerable to damage and is exacerbated by Aβ42 toxicity, or a protective mechanism against both ageing and amyloid toxicity.

The cuticle protein Acp65Aa was also upregulated in Aβ42 flies, but levels fell sharply between 5 and 19 days. However, it is surprising that we identified Acp65Aa in our samples, as it is not expected to be expressed in the brain. One explanation may involve chitin, which has been detected in AD brains and has been suggested to facilitate Aβ nucleation [275]. Amyloid aggregation has previously been shown to plateau around 15 days post-induction [288], which is around the same time that Acp65Aa drops in Aβ42 flies. Our results suggest that Aβ42 causes an increase in Acp65Aa expression early in the disease, but further experiments are needed to confirm this and to investigate its relationship with nucleation and the aggregation process.

2.4.5.2 Hub-bottlenecks: Hsp70A, Gp93, Top2 and Act75B

The four hub-bottlenecks are consistent with Aβ42 inducing stress. Hsp70A, a heat shock protein that responds to hypoxia, was significantly upregulated at early time-points (5 days) in Aβ42 flies, compared to healthy controls which exhibited stable expression of this protein throughout life. Although the levels dropped in Aβ42 flies between days 5 and 31 post-induction, at later time-points Hsp70A increased again, possibly suggesting a two-phase response to hypoxia in Aβ42 flies. We found that Gp93—a stress response protein that binds unfolded proteins—to be increased across age in Aβ42 flies compared to controls possibly suggesting an early and sustained protective mechanism against Aβ42-induced damage. DNA topoisomerase 2 (Top2), an essential enzyme for DNA double-strand break repair, was decreased in Aβ42 flies, following a pattern which mirrors changes in its expression with normal ageing. Double-strand breaks occur naturally in the brain as a consequence of neuronal activity—an effect that is aggravated by Aβ [171]. As a consequence of deficient DNA repair machinery, deleterious genetic lesions may accumulate in the brain and exacerbate neuronal loss.

Finally, we found that actin (Act57B) was increased in Aβ42 flies, in agreement with two recent studies on mice brains [251, 276]. Kommaddi and colleagues found that Aβ causes depolymerisation of F-actin filaments in a mouse AD model before onset of AD pathology [276]. The authors showed that although the concentration of monomeric G-actin increases, the total concentration of actin remains unchanged. It has long been known that G-, but not F-, actin is susceptible to cleavage by trypsin [289], permitting its detection and quantification by IM-DIA-MS. Hence, the apparent increase of actin in Aβ42 flies may be due to F-actin depolymerisation, which increases the pool of trypsin-digestible G-actin, and is consistent with the findings of Kommaddi et al. To confirm whether total actin levels remain the same in the brains of Aβ42 flies, additional experiments would have to be carried out in the future, for example tryptic digestion in the presence of MgADP—which makes F-actin susceptible to cleavage [290]—and transcriptomic analysis of actin mRNA. Furthermore, actin polymerisation is ATP-dependent, so increased levels of G-actin may indicate reduced intracellular ATP. In addition, ATP is important for correct protein folding and therefore reduced levels may lead to increased protein aggregation in AD.

ACSL4, ECHS1, and HSP90B1 have no reported association with AD or related dementias, however, which suggests that our study has the potential to identify new targets in the molecular pathogenesis of this disease. Our study also provides additional information about the homeostasis of these proteins across life from the point of amyloid production. For example, the abundances of Acs1 and Got2 are elevated following Aβ42 induction and continue to increase with age relative to controls. Echs1 is reduced in Aβ42 flies compared to controls but increases across life in parallel with ageing-dependent increases in this protein. Structural proteins Acp65Aa and Act57B are elevated in response to Aβ42, but decline across life whilst remaining stable in control flies. Gp93 and Top2 are either elevated or reduced in response to Aβ42 but mirror ageing-dependent alterations in their expression. mt:CoII is reduced following Aβ42 expression at all time-points, but reduced with ageing in controls. Hsp70A is increased early in Aβ42 flies, reduced to control levels in mid-life, then elevated at late pathological stages whilst remaining stable in healthy controls.

2.4.6 Dysregulated proteins are associated with known AD and ageing network modules

Analysing GO enrichment using network modules, to capture the diverse biological processes modified in AD, we identified 12 modules enriched for processes previously implicated in ageing and AD. This validates the use of our *Drosophila* model in identifying progressive molecular changes in response to Aβ42 that are likely to correlate with progression of cognitive decline in human disease. Further work is required to modify the genes identified in our study at different ages, in order to elucidate whether they represent mediators of toxicity as the disease progresses, factors which increase neuronal susceptibility to disease with age or compensatory protective mechanisms. Model organisms will be essential in unravelling these complex interactions. Our study therefore forms a basis for future analyses that may identify new targets for disease intervention that are specific to age and/or pathological stage of AD.

2.4.7 Conclusion

To understand the dynamic molecular pathology of AD as the disease progresses, we carried out a longitudinal study of the brain proteome using an inducible *Drosophila melanogaster* transgenic model of AD (TgAD) that expresses Arctic mutant Aβ42. We were able to capture how Aβ42 toxicity affects the brain as the disease progresses in the AD brain. We tracked alterations in the brain proteome from the point of amyloid induction, and across life. We identified 3,093 proteins using label-free quantitative ion-mobility data independent analysis mass spectrometry (IM-DIA-MS) [291], 1,854 of which were common to healthy and Aβ42 flies. Of these, we identified 228 proteins that were significantly altered in AD, some of which overlapped with normal ageing but the majority of which were ageing-independent. Proteins altered in response to Aβ42 were enriched for AD processes and have statistically significant network properties in the brain protein interaction network. We also showed that these proteins are likely to be bottlenecks for signalling in the network, suggesting that they comprise important proteins for normal brain function. Our data is a valuable resource to begin to understand the dynamic properties of Aβ42 proteo-toxicity during AD progression. These data may also be useful for researchers to identify potential therapeutic candidates to treat AD at pre- and post-symptomatic stages. Future functional studies will be required to determine the causal role of these proteins in mediating progression of AD using model organisms and to translate these findings to

mammalian systems.

Addendum

During the viva for this thesis, it was brought to our attention that some of the statistical methods that we used to calculate differential gene expression were inappropriate for the type of data analysed in this project. We chose the five methods that we used in this study because they can identify differentially expressed genes in time-course data, albeit in microarray or RNA-Seq data, rather than quantitative mass spectrometry data. The key issue here is that RNA-Seq data are count-based, so should be modelled using discrete distributions, whereas, microarray data and quantitative mass spectrometry data are continuous, so should be modelled using continuous distributions. Discrete data should not be modelled using continuous distributions and vice versa.

EDGE, limma and maSigPro were used appropriately in this study to model differential gene expression using continuous distributions. However, DESeq2 and edgeR were inappropriate for these data, as these methods are based on the (discrete) negative binomial distribution. Interestingly, edgeR, DESeq2 and limma detect similar proteins, whereas, EDGE and maSigPro detect very different proteins (Fig. 2.6).

When we began this work, we were unable to find any methods designed to identify differentially expressed genes in quantitative proteomics data. Subsequently, a number of methods have been developed, for example DEqMS [292], from the same group that developed DESeq and DESeq2, and DEP [293], which is based on limma.

We thank the examiners for bringing this error to our attention.

Chapter 3

Mining metagenomes for new protein functions: applied to plastic hydrolases

3.1 Introduction

3.1.1 Metagenomics and metagenome-assembled genomes

Metagenomics is the study of all genetic material in an environment (biome or microbiome) [294]. Metagenomics typically refers to the study of DNA, whereas, metatranscriptomics refers to RNA. Microbiomes contain a large number of unknown species that have not yet been cultured. It has been estimated [295] that only 1% of microorganisms have been cultured—dubbed 'the uncultured majority' [296]. Many of these species are likely to be eukaryotic [297], whether small or single-celled. Metagenome-assembled genomes (MAGs) are genomes (assemblies) that are recovered from metagenomes by co-assembly of metagenome sequences.

Recently, there has been a growing interest in metagenomics. A number of causes have contributed to this, including sequencing becoming cheaper and easier, improved databases to store metagenomes and metadata, and increasing maturity of bioinformatics tools to process metagenomes. This has allowed microbiomes to be investigated on unprecedented scales, for example to understand the distribution of species in humans [298], the human gut [299, 300], the oceans [301, 302], and distributions of phages across many of the Earth's ecosystems [303]. This has, in turn, bettered our understanding of the interactions between biomes, hosts and microbes, and improve our knowledge of how microbiomes impact host health. The human gut microbiome has been shown to be dominated by environmental factors, rather than host genetics [304]. In other words, cohabiting individuals share microbiomes, whereas, family members who do not live together have

different microbiomes. Furthermore, the gut microbiome has been linked to mental health, quality of life and depression, via the microbiota-gut-brain axis [305].

Metagenomics can be divided up into three steps, outlined in the sections below.

3.1.1.1 Sample collection

Under the metagenomic paradigm, biomes are sampled to collect bulk genetic information, which is analysed *in toto*. Rich metadata are collected alongside genetic material to contextualise results and allow different data sets (possibly collected from the same biome) to be integrated. Examples of metadata include: the type of biome from which samples were collected, such as the human gut, soil, or sea water; the date and location that samples were collected; sequencing platform, data analysis pipeline, and versions of software used; as well as many other esoteric information about particular biomes.

Once a biome has been sampled, a number of further steps are required to prepare the sample to be sequenced [306, 307]. Sample preparation protocols impact the reconstruction quality of metagenomes [308], especially from low biomass biomes, such as skin and soil. Metagenomics studies suffer from large technical variation, which can obscure any meaningful biological variation [309–311]. When technical variations have been controlled for, biological variation has been identified both spatially and temporally [310, 312]. Sample preparation methods contribute greatly to the technical variation. For example, methods of DNA extraction have a large effect on the results of metagenomics studies [313, 314]. Library preparation steps prior to sequencing bias which fragments are sequenced [315]. Efforts are afoot to standardise sample preparation, in order to reduce technical variation [313].

3.1.1.2 Sequencing of samples

Following extraction, genetic material is sequenced to determine the sequence of DNA or RNA. The choice of sequencing platform affects the results of metagenomics studies [311, 316]. Typically, short read Illumina sequencing is performed [317], which typically generates reads < 500 nt. Conceptually, Illumina sequencing is similar to Sanger-type dideoxynucleotide sequencing, but uses fluorescent-labelled nucleosides. Initially, metagenomics was only conducted using amplicon sequencing, where genomic regions of interest were first amplified using libraries of primers before being sequenced. This strategy is cheaper than WGS but only regions that have been selectively amplified are sequenced. On the other hand, WGS is more suitable to metagenomics, where the content of metagenomes

is not known *a priori*, because bulk DNA is sequenced.

Recently, long-read Nanopore sequencing [318] using the MinION sequencer has been gaining in popularity—even being used by PuntSeq in my own backyard to study the metagenome of the river Cam in Cambridge [319]. Compared to Illumina sequencing, MinIONs are small, portable, inexpensive and simple to operate [320], which has democratised sequencing and genomics. In Nanopore sequencing, a polymer is ratcheted through a protein nanopore, which disrupts the ionic current across the nanopore [321]. Polymer sequences can be decoded from the characteristic patterns of currents across nanopores. Nanopore sequencing can even be used to sequence proteins in real-time [322]. Unlike Illumina sequencing, megabase-long contigs can be sequenced in one go using Nanopore sequencing, without needing to be fragmented. The main drawback of the platform is the high error rate, between 5% and 15% [323]. As the errors are uniformly distributed across the length of the sequence sequence, high coverage depth can mitigate errors. Multiple sources contribute to the error rate, including the simultaneous influence of multiple bases on the current across the pore [323]. It is thought that between five and six adjacent bases (corresponding to 4^5 or 4^6 possible k-mers) contribute to the signal at each position in the sequence. Efforts are being made to reduce the error rate, including updates to the sequencing chemistry. Contributions of multiple adjacent bases and a low signal-to-noise ratio produce an enigmatic signal, but computational approaches are being developed to improve the base calling accuracy [323, 324].

Short- and long-read sequencing work synergistically and can be applied in tandem, each alleviating the drawbacks of the other. A key motivation for this is hybrid assembly of reads into contigs [325]. On one hand, short-reads have high accuracy, but are hard to assemble into long contigs. On the other hand, long-reads have low accuracy, but can help to bridge gaps in assemblies to form longer contigs [326].

3.1.1.3 Analysis of samples

Metagenomics produces large volumes of data, on an unprecedented scale for the biological sciences. Processing these data requires new methods, such as MetaSPAdes [327] for genome assembly, MMseqs2 [328] for sequence searches and Linclust [137] for sequence clustering. Performances of various metagenomics tools were compared in a rigorous assessment [329]. MGnify [330] provide a turnkey analysis pipeline for metagenome sequencing reads (https://github.com/EBI-Metagenomics/pipeline-v5) written in Common

Workflow Language [331]. Users need only upload their raw reads to the European Nucleotide Archive [332] to be processed by the MGnify pipeline. MGnify is also a key database for metagenomics data.

Samples are generally analysed in four steps, outlined below.

1. **Assembly of reads into contigs**

   Raw sequence reads are passed through quality control. If necessary, adapter sequences are trimmed. Then, cleaned up reads are assembled into contigs using an assembler, such as MetaSPAdes [327]. Modern assemblers typically split reads up into k-mers and construct a De Bruijn graph [333], $G = (V, E)$. In the graph, vertices represent all $(k-1)$-mers, which are connected by an edge if one vertex is the $[1 : k-1]$ prefix of a k-mer and the other vertex is the $[2 : k]$ suffix [334, 335]. Contigs are assembled from k-mers by finding a Eulerian cycle—a cycle is a path that starts and ends at the same vertex—where each edge is visited once in $\mathcal{O}(|E|)$ time. This strategy was originally devised to answer the Seven Bridges of Königsberg problem, which asked whether a route through the town of Königsberg exists that crosses each of its seven bridges once and only once [336]. Euler devised new methods to prove that this problem has no solution, thus instigating the field of graph theory. Older assembly strategies represented k-mers as vertices [335] and relied on finding a Hamiltonian cycle, where each vertex is visited once. However, finding Hamiltonian cycles is NP-complete [337]—it is a special case of the travelling salesman problem, where the distance between adjacent k-mers is 1, and the distance between all other pairs is 2—so this approach is intractable on anything but small contigs.
2. **Predicting open reading frames**

   Once contigs have been assembled, they are polished to remove errors and scaffolded by joining contigs together [338]. MGnify predicts open reading frames (ORFs) and proteins from assemblies using Prodigal [339] and FragGeneScan [340].
3. **Producing metagenome-assembled genomes**

   Recently, there has been a growing interest in metagenome-assembled genomes (MAGs) [341]. MAGs are recovered from metagenomes, from which many MAGs are assembled, that approximately represent the diversity of a given community. MAGs were pioneered in 2004 by recovering two complete genomes from a biofilm

with low species diversity [342]. Restrictions on the maximum diversity of species in a microbiome have since been lifted [299, 300, 343] by longer read lengths, improved assemblers and better binning algorithms, which has allowed MAGs to be recovered from diverse communities [344]. Binning assigns contigs to a bin if it is likely they are from the same genome. Various properties are used to determine the likeliness, including GC content, tetra-nucleotide frequency and depth of sequencing coverage [344]. MetaBAT is a popular binning algorithm [345]. Binning increases the ability to recover genomes of rare species with low abundance in communities [346]. A Nextflow pipeline for recovering MAGs is available in nf-core [347] (https://github.com/nf-core/mag). MAGs are assigned to a taxonomic group using GTDB-Tk [348].

4. **Assessing the quality of metagenomes**

   Genome-quality, measured by completeness and contamination, can be assessed by checking for presence of universally-conserved genes in particular taxa. Examples of tools include CheckM [349] for prokaryotes, EukCC for eukaryotes [297], and Busco for both [350]. Scores generated by these tools are complementary to classical genome metrics, such as N50 (length of the shortest contig at 50% of the genome length) or L50 (number of contigs that make up 50% of the genome length). Other scores, such as the minimum information about a metagenome-assembled genome (MIMAG) have been devised [344].

3.1.2 The α/β hydrolase domain

α/β hydrolases (ABHs) are hydrolases that contain an ABH domain (Fig. 3.1). The ABH domain fold was first described in 1992 by structural comparisons of proteins that were of very different phylogenetic origin and catalytic function [351]. These proteins, whilst not sharing common sequences or substrates, had a characteristic fold, consisting of an eight-stranded beta-sheet sandwiched between two planes of alpha-helices. A conserved catalytic triad, consisting of nucleophile—histidine—acid residues located on loops between the alpha-helices and beta-sheets, performs catalysis [351]. The triad is reminiscent of the prototypical catalytic triad of serine proteases [352], but the residues are arranged in the mirror inverse—a prototypical example of convergent evolution. Whilst the positions of the catalytic triad's side-chains are exquisitely conserved, the rest of the sequence is not conserved, leading to great structural diversity of the non-core domain structure

[353]. As such, the CATH ABH domain superfamily contains 34 structurally similar groups (5Å complete-linkage clusters), composed of a conserved ABH domain core, that is embelished with many non-conserved structural elements. One consequence of this is that the substrate-binding site is not conserved, permitting ABHs to catalyse multifarious substrates [354, 355]. To show how functionally diverse the superfamily is, its members are annotated with 1,277 unique Gene Ontology (GO) terms and 458 unique Enzyme Commission (EC) terms. The family includes diversely important enzymes, such as acetylcholinesterase, lipase, thioesterase, and various types of peptidases [356]. A number of proteins relevant to biotechnology and pharmaceuticals also contain ABH domains, including many plastic-degrading enzymes, introduced in Section 3.1.4. Rather remarkably, the same ABH can catalyse endopeptidyl hydrolysis and epoxide ring opening, using a single catalytic triad [357].

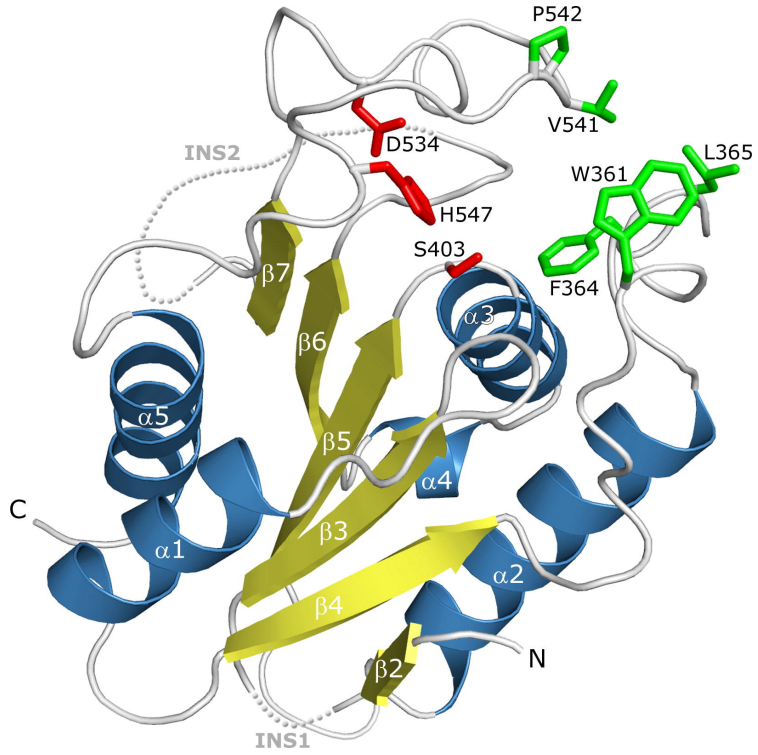


Figure 3.1: **Structure of the α/β hydrolase domain fold.** The β-sheet (yellow) is sandwiched between two planes of α-helices (blue). The nucleophile—histidine—acid (S-H-D in this example) residues (red) are on loops. Figure from [358]. Structure of DUF2319 C-terminal catalytic domain from *Mycobacterium tuberculosis* H37Rv. DUF2319 proteins lack β1 strand of the canonical fold.

The ABH domain fold is represented in CATH as superfamily 3.40.50.1820. In the CATH hierarchy, the ABH domain is in the alpha beta class, three-layer alpha beta alpha

sandwich architecture, and Rossmann fold [359] topology. The ABH domain is one of the largest superfamilies in nature: Gene3D (v16) predicts 557,283 ABH domains in UniProt [49, 360]. The ABH superfamily contains 377 FunFams in CATH (v4.2).

3.1.3 Polyethylene terephthalate (PET)

Plastics are polymers, whose building blocks are derived from crude oil and natural gas. Polyethylene terephthalate (PET) is the most abundant plastic in the world, from its extensive use in packaging and textiles [361]. PET is a polymer of mono(2-hydroxyethyl) terephthalic acid (MHET), which is produced from ethylene glycol and terephthalic acid. Used PET can be recycled by thermomechanical processes, but the resulting material has inferior properties, so a market for virgin PET persists [362]. Globally, 311 million tons of plastics are produced annually, of which PET accounts for 70 million tons [363]. Of concern, only 14% of the plastic produced each year is collected for recycling [363]. A devastating quantity of the remainder ends up polluting Earth, producing untold ecological [364–366] and environmental [367, 368] damage. One poignant example is the not-so-great Great Pacific garbage patch in the North Pacific Ocean [367]: 80,000 tons of plastic in a vile tangle extending over 1.6 million km^2—an area equivalent to the United Kingdom, France, Spain and Germany combined.

Two problems need to be solved to alleviate the impact of plastic on the environment. Firstly, bioremediation methods are required to clean up pollutants. Secondly, to reduce the demand for virgin plastics, recycling methods are required that can convert used plastic into high-quality materials. Both problems could be solved using plastic-degrading enzymes, or organisms.

3.1.4 PET hydrolase enzymes

Naturally-evolved enzymes are able to hydrolase PET. PET degradation has been shown in microbial communities [369] by bacterial hydrolases [370] and cutinases [371, 372]. All known PET hydrolases contain an ABH domain that is responsible for catalytic breakdown of PET. ABHs (Section 3.1.2) are a large superfamily that have evolved a wide variety of functions and PET hydrolases are an exciting biotechnological application that could have a substantial positive impact on the environment. In total, the Plastics Microbial Biodegradation Database (PMBD) lists 79 proteins that are known to be involved in plastic degradation from 949 microorganisms [373]. Furthermore PMBD predicted 8,000 putative plastic-degrading proteins in UniProt. In this chapter, we study the functional diversity of

ABHs in metagenomes, focussing on sequences that are similar to a novel PET hydrolase, PETase.

3.1.5 PETase

In 2016, a new bacterial species was discovered that is able to metabolise PET as its only carbon source [363, 374]. *Ideonella sakaiensis* (*I. sakaiensis*) was isolated from outside a plastic recycling plant in Japan, living on PET bottles—an extreme, unnatural environment, with a high selection pressure to evolve PET metabolism. Two enzymes are responsible: PETase, which depolymerises PET into MHET, and MHETase, which breaks MHET down (Fig. 3.2). The resulting ethylene glycol and terephthalic acid and metabolised by *I. sakaiensis* to provide ATP.

A number of concerns [375] were voiced by Yang *et al.* about the work presented by Yoshida *et al.* [374]. Firstly, Yang identifies that a low crystallinity PET (1.9%) was used to test the efficiency of PETase. Typically, commercial PET bottles have 30 to 40% crystallinity. A cutinase from *Fusarium solanipisi* that hydrolases PET was shown to have a non-linear decrease in PET hydrolysis efficiency as the crystallinity increased [372]. Therefore, PETase was tested on a substrate that is not representative of how the enzyme will most likely be used in the wild. Secondly, Yoshida did not measure the mass of PET to show that it was broken down by *I. sakaiensis*. Instead, they showed a gel permeation chromatogram and presented almost identical traces between a 22 day PETase experiment and a 0 day negative control. Yoshida argue that PET hydrolysis only occurred at the surface of the PET film, but Yang counter argue that breakdown of PET on the surface could be caused by the mechanical action (i.e. not enzymatic) of *I. sakaiensis* on the surface of the film. However, Yoshida argue [376] that their intention was simply to present a microorganism that can grow on PET, and to identify the enzymes that permit this growth. They also argue that microbial degradation of PET had previously been confirmed [369]. In sum, *I. sakaiensis* is able to grow on PET, using PETase, but PETase has not been shown to be able to efficiently hydrolyse the types of PET that are used prolifically in commercial situations.

Whilst PETase and MHETase are responsible for PET degradation [374], the exact mechanisms are unknown. Structural studies have proposed potential mechanisms [377–380]. Despite evolving in a PET-rich environment, PETase does not hydrolyse PET optimally [379]. PETase was modified to narrow the substrate binding cleft, by introducing

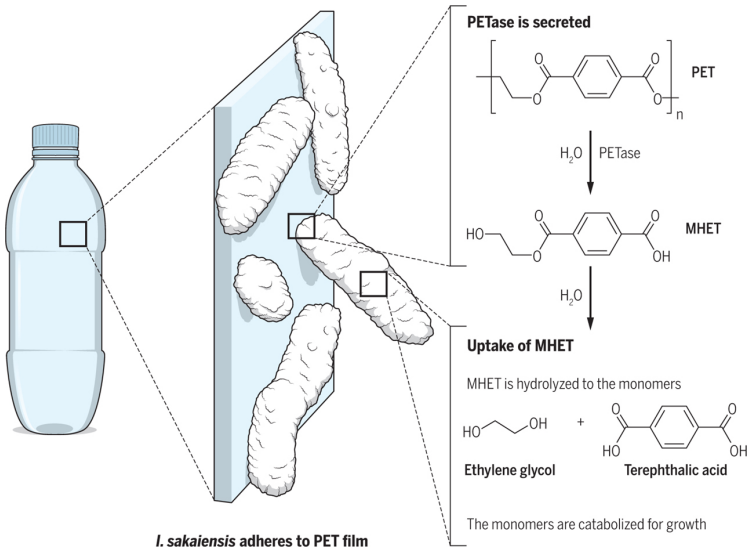


Figure 3.2: **Two enzymes are responsible for PET hydrolysis in *I. sakaiensis*.** PETase depolymerises PET into MHET, followed by MHETase, which breaks MHET down into ethylene glycol and terephthalic acid. Figure from [363].

conserved amino acids that are found in the equivalent positions in cutinases, which improved PET degradation [379]. But this improvement pales in comparison to leaf-branch compost cutinase (LCC) that degrades PET four orders of magnitude faster than PETase [381]. Disulfide bridges were engineered into LCC to make it more thermostable. Post-consumer PET waste was efficiently processed by LCC, with an estimated material cost of required protein at 4% of the ton-price of virgin PET. As a tour de force, PET was then completely recycled using LCC, whereby the breakdown products of ethylene glycol and terephthalic acid, were used to re-form PET, demonstrating the potential for a circular economy [381].

3.1.6 Contributions

The plastic bottle has transitioned from a modern miracle to an environmental scourge, within a generation. Enzymatic bioremediation and recycling of PET is a promising solution to this problem. PET hydrolases are an exciting class of natural enzymes that have recently evolved to degrade PET. Metagenomes are large, untapped reservoirs of novel functions.

In this chapter, we mine metagenomes for ABH domains that may degrade PET or other plastics. We analyse how metagenomic proteins from the MGnify database are distributed between the biomes. Using hidden Markov models (HMMs) of CATH superfamilies from Gene3D and FunFams, we identified $500{,}000$ ABH domains and $400{,}000$ FunFam

domains in MGnify. ABH domains are signficantly enriched in engineered (non-natural) biomes. Metagenomes contain a wide diversity of ABH domains that share only limited overlap with known ABH domains in UniProt. Metagenomic ABH domains share higher sequence similarity with ABH domains from prokaryotes in UniProt, compared to ABH domains from eukaryotes. Many metagenomic ABHs are novel and rare, but those that are common in metagenomes are also found in UniProt. There is little evidence for the evolution of novel domains in metagenomic proteins that contain an ABH domain. Metagenomic ABH domains are distributed amongst the ABH FunFams in a completely different pattern to how ABH domains from UniProt are distributed.

Large sequence data sets are becoming increasingly common with the advent of large-scale genome sequencing and metagenomics. GARDENER, the current method for generating FunFams is unable to scale to data of this size. In this chapter, we also develop a new divide-and-conquer algorithm, FRAN, to generate FunFams on large sequence data sets. We rigorously benchmark the performance of FRAN and compare it to GARDENER.

3.2 Methods

3.2.1 Data

3.2.1.1 MGnify

MGnify [330] is a microbiome database. MGnify hosts metagenome sequences from microbiome studies. Typically, uploaded data sets are from shotgun metagenomics [382]. MGnify assembles reads into contigs using metaSPAdes [327], a De Bruijn graph assembler. With contigs longer than 500 nucleotides in hand, RNA genes are predicted using Rfam [383] and masked. ORFs are predicted in the remaining sequences using the prokaryotic gene callers Prodigal [339] and FragGeneScan [340]. Prodigal is run first, followed by FragGeneScan for regions in which Prodigal does not predict any proteins. Protein sequences were clustered using Linclust at 90% sequence identity with 90% coverage of the centroid sequence (command line arguments `--min-seq-id 0.90 -c 0.9 --cov-mode 1`).

Metagenome studies are associated with metadata. One such datum is the biome from which the metagenome was sampled. Biomes are classified according to the Genomes On-Line Database (GOLD) [384] microbiome ontology, which is a directed acyclic graph that describes hierarchical relationships between biomes, akin to the Gene Ontology for func-

tion annotations. Whilst biome classification at the study level is mutually exclusive, it is not so at the protein sequence level because the same sequence can be found in multiple biomes. A selection of 13 high-level ontological terms are associated with sequences (demarcated by *):

- Engineered*
- Environmental
  - Aquatic*
    - Marine*
    - Freshwater*
  - Soil*
    - Clay*
    - Shrubland*
- Host-associated
  - Plants*
  - Human*
    - Digestive system*
  - Human but not root
    - Digestive system*
  - Animal*
- None of the above*

For reference, UniProt contains 181,252,700 sequences, comprised of 562,253 Swiss-Prot sequences and 180,690,447 TrEMBL sequences (Fig. 3.3). Here, we used MGnify May 2019 release, which contains 1,106,951,200 protein sequences (Fig. 3.3), generated from 12,560 assemblies. A non-redundant set of 304,820,129 is available as S90 cluster representatives. From now on, we refer to the non-redundant sequences as 'MGnify proteins'. MGnify contains vastly more sequences than UniProt, so is likely to contain novel functions yet to be discovered.

3.2.1.2 CATH, Gene3D and FunFams

CATH, Gene3D and FunFams were introduced in Sections 1.3.4, 1.3.4.1 and 1.3.4.2. CATH [33] v4.2 and Gene3D [49] v16 were used. Superfamily HMMs in Gene3D v16 were generated using UniProt [360] August 2017 release. Gene3D contains models for 6,119 superfamilies, represented by 65,014 HMMs trained on multiple sequence alignments of S95

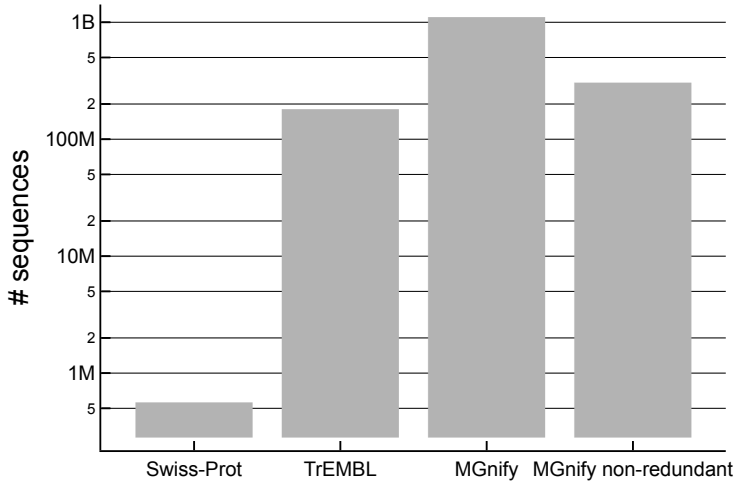


Figure 3.3: **Number of sequences in Swiss-Prot, TrEMBL and MGnify.** MGnify non-redunant are S90 cluster representative sequences.

sequence identity clusters. ABH domain sequences predicted by Gene3D from UniProt August 2017 release were used.

3.2.2 Software

3.2.2.1 Containers

Singularity [385] v2.6.0 was used to run Singularity and Docker containers. Docker (https://www.docker.com) images were downloaded from Docker Hub (https://hub.docker.com) and BioContainers [386].

3.2.2.2 HMMER

HMMs and HMMER were introduced in Section 1.3.2. HMMER [27] v3.2.1 was used from the Docker image "biocontainers/hmmer:v3.2.1dfsg-1-deb_cv1".

3.2.2.3 Linclust

Linclust [137] is a fast, greedy, single-linkage sequence clustering method in MMseqs2 [328]. Due to some clever tricks, Linclust's runtime scales linearly with the number of sequences, independently of the number of clusters. First, Linclust performs an approximate and inexpensive clustering by assigning each sequence to multiple sets, or canopies [387]. Canopy clustering approaches improve time complexities by applying exact but expensive clustering methods to each canopy independently. Linclust uses sequence MinHashing [133–135, 388, 389] to generate canopies. p different hash functions are used to select the p k-mers from each sequence with the lowest hash value. Here, we used $p = 21$.

Sequences that share a k-mer are assigned to the same canopy. The longest sequence in each canopy is selected as the centroid sequence. Edges are added between centroids and the sequences in the canopy to form an initial single-linkage clustering. Expensive comparisons, such as sequence alignment, are only performed between $\leq pn$ connected sequence pairs. $\mathcal{O}(pn)$ runtime complexity is achieved where $p << n$, compared with $\mathcal{O}(n^2)$ for all-against-all comparison. Finally, edges are removed between sequence pairs if one or more clustering criteria, such as the sequence identity threshold, are not met.

Linclust v10 was used from the Docker image "soedinglab/mmseqs2:version-10". Sequences were clustered at different sequence identity thresholds and coverage of centroid sequences, typically 30%, 50%, 70% or 90%, (command line arguments: `--min-seq-id <X> --c <X> --cov-mode 1`, where X is some percentage, provided as a fraction).

3.2.2.4 cath-resolve-hits

cath-resolve-hits was introduced in Section 1.3.4.1. cath-resolve-hits [52] v0.16.2 was used to resolve domain boundaries in multi-domain architectures (MDAs). Different parameters were used depending on the context, so these are defined in the protocols. cath-resolve-hits was used from the Docker image "harryscholes/cath-resolve-hits:0.16.2".

3.2.2.5 Nextflow

Nextflow [390] (v19.07.0.5106) was used to write and execute reproducible pipelines. Nextflow introduces a number of useful features for deploying pipelines, particularly in high-performance computing cluster environments. Nextflow pipelines are constructed from building blocks of 'processes', arranged in linear or branched topologies. Each process contains a set of inputs, a script that processes the inputs, and a set of outputs. The script can be in any language, be it Bash, Julia or Python. Parallelism can be easily introduced by splitting inputs into chunks, to be processed separately. Processes only request their required CPU and memory resources from cluster queue managers. To achieve true portability, each process can be executed in a container (Docker and Singularity are supported), which can be automatically pulled from various container repositories. Intermediate results are cached, which allows pipelines to be resumed and updated without repeating a lot of computation. Many expert-written pipelines are available from nf-core [347], a community effort to provide complex bioinformatics pipelines to users, requiring little (computational) domain-specific knowledge. In sum, Nextflow is one of many new

tools, which make it easy to write and execute reliable and reproducible pipelines.

Where appropriate, pipelines described in this work are implemented in Nextflow. These pipelines are designed to be portable and reproducible. For example, input data are automatically downloaded where possible, processes are executed in Singularity containers, and data are processed in parallel for speed.

3.2.3 Data analysis

Data analyses were performed in Julia v1.3 [391] using packages BioAlignments v2.0.0, BioSequences v2.0.1, CodecZlib v0.7.0, Combinatorics v1.0.0, CSV v0.6.1, DataFrames v0.20.2, DataStructures v0.17.11, FASTX v1.1.1, Formatting v0.4.1, GLM v1.3.9, HTTP v0.8.13, HypothesisTests v0.9.2, JSON v0.21.0, JSON2 v0.3.1, KernelDensity v0.5.1, Plots v1.0.5, StatsPlots v0.14.4, TranscodingStreams v0.9.5 and Unmarshal v0.3.1. Custom library code is released as CATHBase.jl [392], which is available from https://github.com/harryscholes/CATHBase.jl.

3.2.4 Finding domains in protein sequences

In this context, we define a domain as a region of a protein sequence that has a significant hit to an HMM. The following protocol can be used to find domains in a database of protein sequences:

1. Search the sequence database using a library of domain HMMs
2. For each protein:
    i. Resolve the MDA (optional)
    ii. Extract domains

This is a generic and flexible protocol to find domains in protein sequences. Below, we set out specific protocols that we used in this study to find CATH superfamily and FunFam domains in the MGnify protein database.

3.2.4.1 Predicting CATH superfamily domains in protein sequences

CATH superfamily domains were predicted in protein sequences using Algorithm 3.1. The MGnify protein database was searched using the Gene3D HMM library using hmmsearch from HMMER3 [27]. Hits were called using a threshold of $\mathrm{E} < 0.001$ (command line arguments `--domE 0.001 --incdomE 0.001`). It is often expedient to split the sequence database into manageable chunks, which can be searched independently and in parallel. For E-values to be calculated correctly, the size of the entire database must be provided to the `-Z` command line argument.

Algorithm 3.1 Predict CATH superfamily domains in protein sequences.

1: **procedure**
2: scan sequences against the Gene3D HMM library
3: resolve MDAs using cath-resolve-hits
4: extract superfamily domains of interest from the resolved MDA coordinates
5: **end procedure**

cath-resolve-hits was used to resolve MDAs (command line arguments `--min-dc-hmm-coverage=80 --worst-permissible-bitscore=25`). CATH superfamily IDs were assigned using the `assign_cath_superfamilies.py` Python script provided with Gene3D. If all domains were continuous, this process would be trivial, however, this is not the case and discontinuous domains complicate the process. Discontinuous HMMs are made up of multiple HMMs that model the separate continuous sequences. It is trivial to assign the MDA of a protein containing all continuous domains. For proteins that contain discontinuous domains, there is no correct way to assign the MDA string. Here, for a domain D in protein P, we calculated its centre of mass $\mathbf{R}$ by

$$\mathbf{R}(D|P) = \frac{\sum D_i}{\sum P_j}$$

for all i residue numbers in D and all j residue numbers in P. MDAs were assigned by sorting domains according to their centre of mass.

ABH domain sequences were extracted from full-length sequences using the resolved start and stop positions of continuous domains and the concatenated sequence of discontinuous domains from the myriad start and stop positions.

3.2.4.2 Predicting CATH superfamily domains in large sequence data sets

Algorithm 3.2 Predict CATH superfamily domains in large sequence data sets.

1: **procedure**
2: scan sequences against a subset of the Gene3D HMM library
3: create a subset of the sequences with a hit to one of the HMMs
4: Algorithm 3.1 continues from Line 1
5: **end procedure**

In actuality, we used a modified version of Algorithm 3.1 to search for superfamily domains in the 300 million MGnify proteins (Algorithm 3.2). To reduce the amount of computation, we added additional steps. These steps are only appropriate if searching for a subset of domains.

First, the sequence database was searched using a subset of models. In this case, the MGnify proteins were searched using the Gene3D HMMs from the ABH superfamily and the same settings as Section 3.2.4.1. Second, the sequences that have hits to the subset of models are extracted to create a derivative database that is likely to be much smaller than the original database. The derivative database is used as input to the protocol in Section 3.2.4.1. This modified protocol is implemented as a Nextflow pipeline.

3.2.4.3 Predicting FunFam domains in protein sequences

FunFam domains were identified in sequences that contain ABH superfamily domains from Section 3.2.4.2 using Algorithm 3.3. These sequences were scanned against the FunFam HMM library using HMMER. The per FunFam inclusion threshold was used (command line options `--cut_tc`). MDAs were resolved using cath-resolve-hits (command line options `--min-dc-hmm-coverage=80 --worst-permissible-bitscore=25 --long-domains-preference=2`). This protocol is also implemented as a Nextflow pipeline.

Algorithm 3.3 Predict FunFam domains in protein sequences.

1: **procedure**
2: scan sequences against FunFam HMMs
3: resolve MDAs using cath-resolve-hits
4: extract FunFam domains of interest from the resolved MDA coordinates
5: **end procedure**

3.2.5 Protein sequence clustering

ABH domains from full-length MGnify sequences and UniProt were pooled to create a combined ABH domain data set. These sequences were clustered into single-linkage clusters using Linclust at 30%, 50%, 70% and 90% sequence identity and coverage of the cluster centroid sequence, implemented as a Nextflow pipeline.

3.2.6 Kingdom-level taxonomy of UniProt ABH domains that cluster with MGnify ABH domains

Mixed clusters, composed of ABH domains from MGnify and UniProt, from clustering at S30, S50, S70 and S90 (Section 3.2.5) were used. The kingdom-level taxonomy of UniProt sequences in mixed clusters were downloaded using the Proteins API [393].

3.2.7 Identifying whether novel domains have evolved in metagenomes

C- and N-terminal regions and inter-domain regions of sequence may contain novel domains that evolved in metagnomes. To test the evidence for novel domain evolution, terminal and inter-domain sequence lengths were calculated from Gene3D hits (Section 3.2.4.2). Inter-domain regions are defined as contiguous regions of sequence that do not have a significant match to any Gene3D HMM in the protein's MDA. Full-length proteins whose MDAs only contain continuous domains and do not contain discontinuous domains were considered. Single-domain proteins have two terminal regions, whereas, multi-domain proteins can also have inter-domain sequences between each pair of adjacent domains. Examples are shown below for domains (@) and terminal and inter-domain regions (-).

i. Single-domain protein:

```
          -----@@@@@@@-----
Terminal:   1             2
```

ii. Multi-domain protein:

```
          -----@@@@@@@---@@@@@@@---@@@@@@@-----
Terminal:   1                                 4
Inter-domain:             2         3
```

3.2.8 Identifying redundant domain sequences

Full-length sequences were aligned pairwise by the banded Needleman-Wunsch algorithm [41] for global sequence alignment from BioAlignments.jl. A constant gap penalty was used with -1 penalty for opening a gap and no penalty for extending a gap. The BLOSUM62 substitution matrix was used [394]. Alignment scores were calculated as

$$A_{S_1S_2} = \sum_m + \sum_g$$

where m are the substitution matrix scores for each match and mismatch, and g are the gap opening penalties. Distances D were calculated from alignment scores [395] by

$$D_{S_1S_2} = 1 - \frac{A_{S_1S_2}}{\min(A_{S_1S_1}, A_{S_2S_2})}.$$

3.2.9 Biome distribution of metagenomic α/β hydrolases

For each of the 13 biomes, the number of protein sequences was calculated. The number of ABH domains that were found in these sequences was also calculated.

3.2.10 Algorithms to generate FunFams on arbitrarily large numbers of sequences

Currently, FunFams are generated by GeMMA building a tree from all S90 starting clusters of an MDA (Section 1.3.4.2). This is disadvantageous because GeMMA has $\mathcal{O}(n^3)$ time complexity (Tony Lewis personal communication). Naive hierarchical clustering is $\mathcal{O}(n^3)$, but it can be improved to $\mathcal{O}(n^2)$ [396]. Given n starting clusters, GeMMA performs $\mathcal{O}(n)$ merges, in which the $\mathcal{O}(n^2)$ distance matrix is searched before each merge, producing $\mathcal{O}(n^3)$ overall. FunFHMMer then partitions the GeMMA tree into FunFams in $\mathcal{O}(n^2)$ time (Tony Lewis personal communication). In practical terms, however, calculating the distance matrix is much more costly than searching the distance matrix—an example of where there is a mismatch between complexity analysis and empirical runtimes. Calculation of the distance matrix involves aligning sequences in the newly merged cluster, training an HMM on the alignment, and then aligning this HMM with HMMs of other clusters.

To cope with memory limits, we currently:

- Generate FunFams within an MDA partition. Proteins with different MDAs are unlikely to be in the same FunFam, so it is reasonable to segregate them *ab initio*.
- Only generate FunFams from S90 clusters that have at least one experimental GO term annotation—these are known as the starting clusters. In so doing, each FunFam is associated with at least one high-quality function. However, a scalable protocol would allow all S90 clusters to be used to generate FunFams, without discarding S90 clusters that do not have any experimentally characterised functions. In doing so, FunFams would be able to capture sequences that have novel functions, rather than only representing functions that have already been experimentally characterised. Consequently, some FunFams will not have any associated functions, but newly characterised functions could be attached to the respective FunFams on the fly.

This is all well and good, but some MDA partitions already have a prohibitively large number of starting clusters. So, a new method to generate FunFams at gigascale must be found.

Whilst GARDENER (Section 1.3.4.3 and Fig. 1.5) improves the quality of FunFams, it

still exceeds memory limits because all sequences must be available at all times. However, GARDENER did show that it is possible to merge FunFams by treating them as starting clusters. We use this knowledge in designing a method to generate FunFams at gigascale.

An aside on scaling FunFams There may be a low-hanging fruit solution to the FunFam generation problem that will not be tackled in this chapter. Currently, GeMMA grows a tree from leaves to root. FunFHMMer then partitions the tree into FunFams. GeMMA trees are cut closer to the leaves than the root, so the later node merges are redundant. Although there are fewer of them, later merges are much more costly than early ones. The low-hanging fruit is to only grow a partial tree. To do this, FunFHMMer would be run in concert with GeMMA, every k merges, where k is sufficiently large. GeMMA would be stopped prematurely if the FunFam criteria are met. Running GeMMA and FunFHMMer in serial is $\mathcal{O}(n^3) + \mathcal{O}(n^2)$, which reduces to $\mathcal{O}(n^3)$ in big O notation. If the entire tree is grown, the iterative procedure proposed here is $\mathcal{O}(n^3) + \mathcal{O}(\frac{n}{k}n^2)$ in the worst case when FunFHMMer is run $\frac{n}{k}$ times, which also reduces to $\mathcal{O}(n^3)$. On average, the empirical runtime will be much less than this because the FunFam criteria will be met before the entire tree is grown. We were interested in implementing this new algorithm, but time constraints and lack of Perl knowledge prevented it.

3.2.10.1 Divide-and-conquer strategies

A different approach to let GeMMA scale to large sequence databases is to employ a divide-and-conquer strategy. Computer science has a long history of success with improving the runtime complexity of algorithms, with worse than linear complexity, using divide-and-conquer approaches and parallelism. Sorting is a prototypical example of divide-and-conquer and its benefits. A brute force sorting implementation takes $\mathcal{O}(n^2)$ operations to compare all elements. Divide-and-conquer implementations have better performance: $\mathcal{O}(n \log n)$ for merge sort, and $\mathcal{O}(n \log n)$ expected complexity for quicksort.

Can we generate FunFams using a divide-and-conquer approach? To do so, we would have to be able to merge the results of the subproblems to form the final set of FunFams. One possible algorithm is Algorithm 3.4.

By using the FunFams from each subproblem as starting clusters for a second round of FunFamming, sequences that should be in the same FunFam, but were sampled into different groups, will be allowed to merge.

Parsimoniously, starting clusters can be divided into groups by random sampling.

Algorithm 3.4 Divide-and-conquer algorithm to generate FunFams.

```
1: procedure
2:     generate starting clusters for a set of sequences
3:     divide starting clusters into groups
4:     for group in groups do
5:         generate FunFams of each group using GARDENER
6:     end for
7:     pool FunFams and treat them as starting clusters
8:     generate FunFams using GARDENER
9: end procedure
```

However, this is unlikely *a priori* to produce informative FunFams. For FunFHMMer to partition the GeMMA tree into informative FunFams, a sufficient level of sequence diversity is required to highlight the specificity-determining positions. Superfamily domain sequences are evolutionarily related, so this information should be taken into account when dividing starting clusters into groups. Conceptually, this is akin to the current use of evolutionary MDA information to divide sequences.

Let M be the set of protein sequences from a superfamily that have the same MDA. M is also known as an MDA partition. The goal is to subset M into k representative groups, G_k, each of size n. To do this, sequences are sampled into groups, such that certain characteristics of M are preserved. Namely, we wish G_i to have approximately the same sequence diversity as M, whilst not being biased to any dominant regions of sequence space. In designing this algorithm, we were inspired by two clever algorithms, one for clustering and another for sampling, which we now explain briefly. Canopy clustering [387] initially clusters items into sets, or canopies, using a cheap method, followed by expensive clusterings of items within each canopy. The second method, geometric sketching [397], uses by 'geometric' sampling to summarise large data sets using a small subset of the data that preserves the geometry, rather than the density, of the whole data set. Initially, a high-dimensional data space is covered with a lattice of equal-sized boxes, from which sketches are generated by uniformly sampling a box, followed by uniformly sampling an item from that box, repeated until the sketch is the desired size.

Thus, we designed the following algorithms:

3.2.10.2 FRAN

FunFam generation by **ran**dom sampling (Algorithm 3.5), a uniform random sampling method. FRAN generates groups by uniform random sampling of S90 clusters, without

attempting to preserve the desirable characteristics of M. Thus, FRAN is the baseline performance for FunFam generation on random subsets of M.

Algorithm 3.5 FRAN.

$M \leftarrow$ MDA partition
$k \leftarrow$ desired number of groups
procedure FRAN(M, k)
    $n \leftarrow M$`.size()` / k
    $G_k \leftarrow$ initialise groups
    $C \leftarrow$ cluster M into S90 clusters
    for $i \leftarrow 1...k$ **do**
        while G_i`.size()` $< n$ **do**
            $x \leftarrow$ uniformly sample an S90 from C without replacement
            G_i`.append(`x`)`
        end while
    end for
end procedure

3.2.10.3 $\text{FRAN}_{\text{geometric}}$

FunFam generation by **ran**dom **geometric** sampling (Algorithm 3.6), a geometric sampling method. The difference in performance between $\text{FRAN}_{\text{geometric}}$ and FRAN will demonstrate the added value of considering the evolutionary relationships between sequences to maintain the desirably characteristics of M. In practice, we actually implemented Algorithm 3.6 as Algorithm 3.7.

Algorithm 3.6 $\text{FRAN}_{\text{geometric}}$.

$M \leftarrow$ MDA partition
$k \leftarrow$ desired number of groups
procedure $\text{FRAN}_{\text{GEOMETRIC}}(M, k)$
    $n \leftarrow M$`.size()` / k
    $G_k \leftarrow$ initialise groups
    $C \leftarrow$ cluster M into S30 clusters
    for all C_i **do**
        $D_i \leftarrow$ cluster C_i into S90 clusters
    end for
    for all G_i **do**
        while G_i`.size()` $< n$ **do**
            $x \leftarrow$ uniformly sample an S30 from C
            $y \leftarrow$ uniformly sample an S90 from x without replacement
            G_i`.append(`y`)`
        end while
    end for
end procedure

Algorithm 3.7 FRAN$_{\text{geometric}}$ implementation.

```
1:  M ← MDA partition
2:  k ← desired number of groups
3:  procedure FRAN_GEOMETRIC(M, k)
4:      n ← M.size() / k
5:      G_k ← initialise groups
6:      C ← cluster M into S90 clusters
7:      D ← cluster C S90 cluster representatives into S30 clusters
8:      for all C_i do
9:          D ← cluster C_i into S90 clusters
10:         for j ← 1...k do
11:             while G_j.size() < n do
12:                 x ← uniformly sample an S30
13:                 y ← uniformly sample an S90 from x without replacement
14:                 G_j.append(y)
15:             end while
16:         end for
17:     end for
18: end procedure
```

Algorithm 3.7, line 6 is $\mathcal{O}(|M|)$, whilst line 7 is $\mathcal{O}(|C|)$, where $|C| < |M|$. These time savings can be made because each cluster representative represents the sequence diversity of its cluster. Each group G_i then takes the place of M in the extant FunFam algorithm, GARDENER. As sequence data sets continue to grow in size, FunFam generation can be made computationally tractable once more because $|G_i| << |M|$.

FRAN and FRAN$_{\text{geometric}}$ were benchmarked against GARDENER (Section 1.3.4.3)—the current method to generate FunFams, which we considered to be the gold standard. FunFams were generated in the usual way for GARDENER, and for FRAN and FRAN$_{\text{geometric}}$ using Algorithm 3.8. NB because the majority of ABH domains occur in single-domain MDAs, GARDENER consisted of a single round of GeMMA and FUnFHM-Mer with all MDAs combined. Sequences from the CATH v4.3 FunFam seed alignments, rather than the full alignments, were used (Section 1.3.4.3).

The methods were benchmarked by generating FunFams of two CATH superfamilies: α/β hydrolase fold, catalytic domain (3.40.50.1820) and Phosphorylase Kinase; domain 1 (3.30.200.20). For FRAN and FRAN$_{\text{geometric}}$ the starting clusters were sampled into $k = 3$ groups. For each FunFam, F, we measured the performance of the three methods using three metrics:

i. EC purity: $\text{EC purity}(F) = \frac{|\text{sequences with most common EC term}|}{|\text{sequences with any EC term}|}$.

Algorithm 3.8 FRAN and FRAN$_{\text{geometric}}$ benchmark.

```
procedure
    G ← sample starting clusters into groups of approximately the same size
    for all G_i do
        run GeMMA
        run FunFHMMer
    end for
    pool the |G| FunFHMMer outputs and use as inputs to GeMMA
    run GeMMA
    run FunFHMMer
end procedure
```

ii. Rand index: Let $S = \{o_1, ..., o_n\}$ be a set of n elements. The Rand index R can be used to compare two clustering methods that partition S into k subsets, $X = \{X_1, ..., X_k\}$, and l subsets, $Y = \{Y_1, ..., Y_l\}$, respectively. The Rand index is defined as

$$R = \frac{a + b}{\binom{n}{2}}$$

where

- a is the number of pairs of elements in S that are in the same cluster in X and Y,
- b is the number of pairs of elements in S that are in different clusters in X and Y.

$R = 1$ if and only if $X = Y$. $R = 0$ if elements are clustered completely differently in X and Y. Rand indexes were calculated to compare FunFam clusterings, whereby only FunFams that were composed of two or more starting clusters were considered.

iii. Graph theoretic measures: Let $S = \{S_1, ..., S_k\}$ be a set of starting clusters that are clustered into a set of FunFams $F = \{F_1, ..., F_l\}$ be a set of FunFams by some method M. Let $G = (S, E)$ be a graph of S where pairs of starting clusters are connected by an edge if M clusters them into the same FunFam (Fig. 3.4). Therefore the l FunFams are both the maximal cliques and connected components of G. A clique is a subgraph, where every pair of nodes are adjacent. A maximal clique is a clique that cannot be extended, to form a larger clique, by including one adjacent node.

FunFam graphs were constructed for FunFams generated by FRAN, FRAN$_{\text{geometric}}$ and GARDENER. For all combinations of these graphs, a new graph $G_u = G_1 \cup G_2$ was constructed from the union of nodes and edges (Fig. 3.4). The number of maximal cliques and connected components in G_u were calculated. If the FunFams in G_1 and G_2 agree, the number of maximal cliques and connected components will be the same. However,

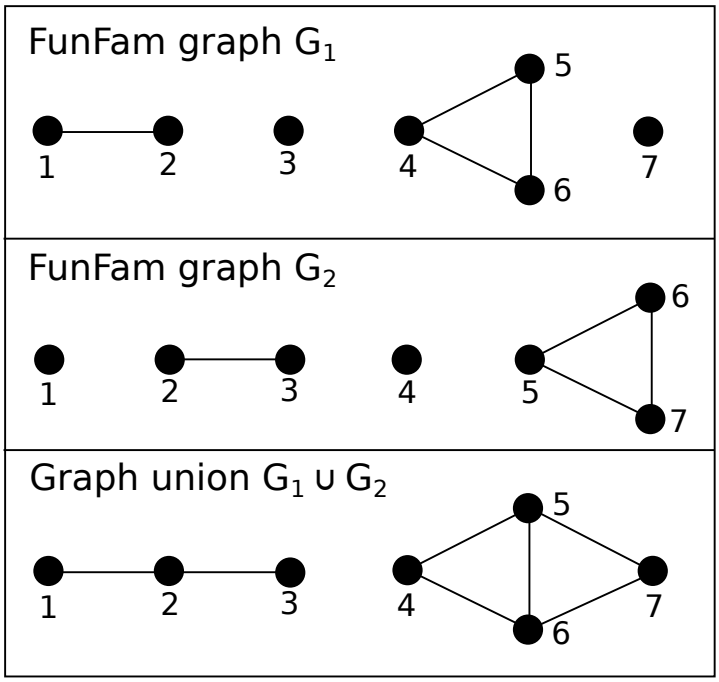


Figure 3.4: **FunFam graphs and their graph unions.** Nodes in FunFam graphs are starting clusters that are connected by an edge if they are clustered into the same FunFam. FunFams are both the maximal cliques and connected components in FunFam graphs. Graph unions G_u can be constructed from the node and edge sets of two FunFam graphs G_1 and G_2.

if the FunFams in G_1 and G_2 partially agree, the number of connected components will decrease, whilst the number of maximal cliques will increase. This will occur when a pair of FunFams share at least one, but not all, starting clusters. As such, maximal cliques that were separate connected components are now connected in the same component.

3.3 Results

3.3.1 Biome distribution of proteins found in metagenomes

We assessed which biomes the metagenomic protein sequences were sampled from (Fig. 3.5). We used the S95 cluster representatives of MGnify's predicted protein sequences. The most populous biome is 'Aquatic' with 142,232,832 sequences. This is followed by 'Marine', a sub-biome of 'Aquatic', with 104,242,612 sequences, which corresponds to 73.3% of sequences in the 'Aquatic' biome. The 'Engineered' biome contains the third most sequences at 78,057,115. Engineered biomes encompass a broad range of non-natural biomes, including industrial settings, laboratory conditions or waste treatment. Fourth is 'Human' with 67,475,113 sequences, followed by the human 'Digestive system' sub-

biome with 66,357,615 sequences, corresponding to 98.3% of sequences in 'Human'. This may be caused by the large bias within metagenomics for sampling the human gut. There are no protein sequences in MGnify from the biomes 'Animal' or 'Soil', or any of its sub-biomes 'Clay' and 'Shrubland'. This is a consequence of which metagenome samples have been assembled in MGnify, for ORFs to be predicted in. Assembly is extremely expensive and slow, so is a rate-limiting factor in predicting proteins from all metagenome samples in MGnify. In the future, many more samples, from a wide assortment of biomes, will be assembled.

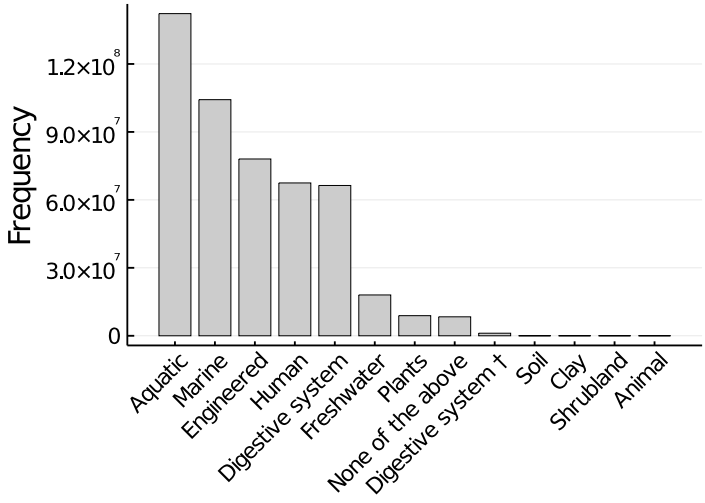


Figure 3.5: **Biome distribution of MGnify protein sequences.** Biomes are ordered by number of sequences. † = 'Host but not root:Digestive system'.

3.3.2 Metagenomes contain many α/β hydrolase domains

We searched for α/β hydrolase domains in metagenomes using Gene3D and FunFams.

3.3.2.1 ABH superfamily domains

To identify CATH superfamily domains, we scanned the MGnify protein sequences against the Gene3D models for ABHs (superfamily ID 3.40.50.1820). To reduce the sequence redundancy of the database, we used the 304,820,129 MGnify protein sequences. We used the modified protocol to find domains in large protein sequence data sets described in Section 3.2.4.2. First, the sequences were scanned against the 416 Gene3D HMMs for the ABH superfamily. 1,630,166 sequences contained ABH domain hits with a significant E-value. These sequences were scanned against Gene3D HMMs for all superfamily domains. 1,942,889 domain hits were found for all CATH superfamilies, of which 1,450,276 were

ABH domains consisting of 1,444,433 were unique ABH domain sequences. Following assignment of MDAs, 1,435,764 sequences contained ABH domains, whilst 194,402 sequences were subsequently found to not contain ABH domains.

Prodigal was used to predict whether sequences are full-length, truncated at the N-terminus, C-terminus, or both (Fig. 3.6). 508,693 proteins (35%) are predicted to be full-length. 46% of sequences were truncated at one end, split equally between 330,328 N-terminal and 330,043 C-terminal truncations. Finally, 266,700 (19%) of sequences were truncated at both ends. Due to the size of the data, we only took full-length sequences, and any ABH domains contained in full-length sequences, forward for further analysis.

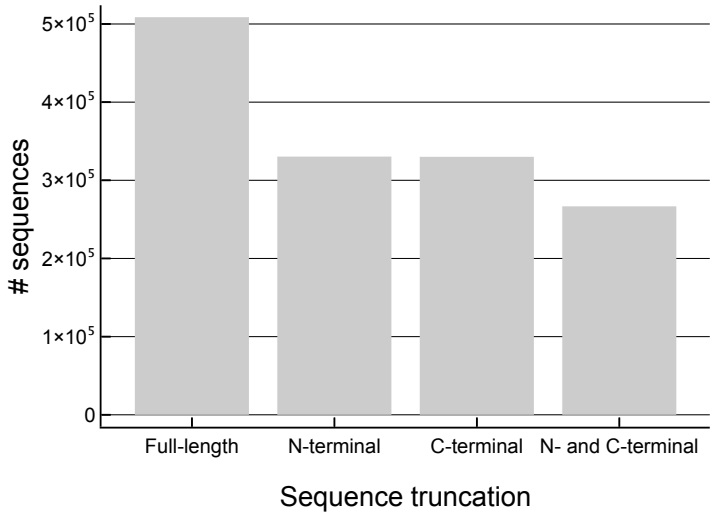


Figure 3.6: **Truncation of MGnify protein sequences.** Prodigal was used to predict whether sequences are full-length, truncated at the N-terminus, C-terminus, or both.

Lengths of ABH domains from different databases were compared. Length distribution of ABH domains from MGnify agree well with those from UniProt and Gene3D HMMs (Fig. 3.7). Gene3D ABH HMMs, built from structures in CATH v4.2, have median length 284 residues. ABH domains in UniProt sequences from Gene3D have median length 259. It is reasonable to expect that the median match length will be shorter than the median HMM length because subsequences can match HMMs with significant E-values. ABH domains in MGnify have median length 247 residues.

3.3.2.2 ABH FunFam domains

Having found MGnify proteins that contain ABH domains, we next wanted to identify to which ABH FunFams these domains match. We scanned the full-length sequences that

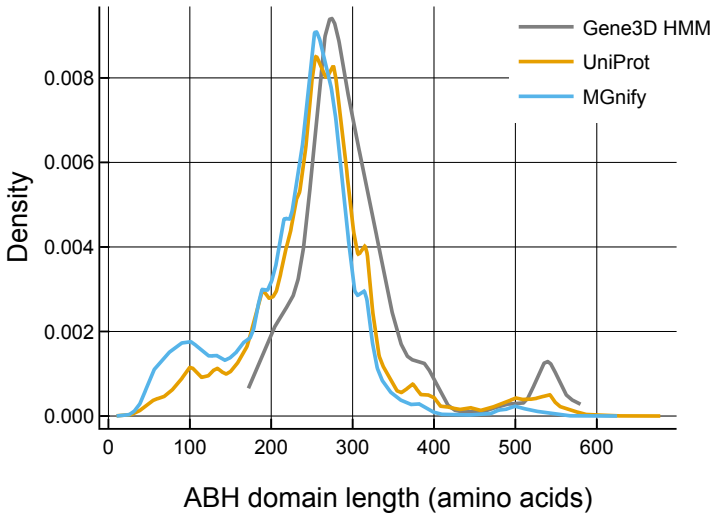


Figure 3.7: **Distribution of ABH domain lengths.** Lengths of ABH domains from MGnify, UniProt and Gene3D HMMs are plotted. The probability density function of each distribution was estimated using kernel density estimation.

contain the 508,693 ABH domains against the FunFam HMM library. Whilst CATH superfamily domains are assigned using a lenient significance threshold of $E < 0.001$ in Gene3D, sequences are assigned to FunFams using a much stricter threshold, known as the FunFam inclusion threshold. An inclusion threshold is generated by scanning sequences from a FunFam alignment against the FunFam HMM. The lowest (worst) bit score is the inclusion threshold. There may be many and overlapping FunFam matches to a sequence. A researcher may wish to know all FunFam matches, in which case these matches are the desired output. Instead, if a researcher prefers to know an MDA, then cath-resolve-hits can be run to resolve the matches to the optimal set of non-overlapping FunFams.

After resolving FunFam MDAs, we found 398,580 significant FunFam hits in 360,119 sequences. Of these, there were 357,073 hits to ABH FunFams, in 351,853 protein sequences. There is not a one-to-one mapping between ABH hits from Gene3D and FunFams. Some proteins did not have any FunFams: 148,574 proteins that were predicted to have an ABH domain do not have any hits to FunFams from any superfamily. Other proteins did not have any ABH FunFams: 156,840 proteins that were predicted to have ABH domains do not have any hits to ABH FunFams.

Lengths of ABH FunFam matches are distributed similarly to superfamily matches (Fig. 3.8). Large peaks for FunFam HMMs and FunFam matches exist at a length of 260 amino acids.

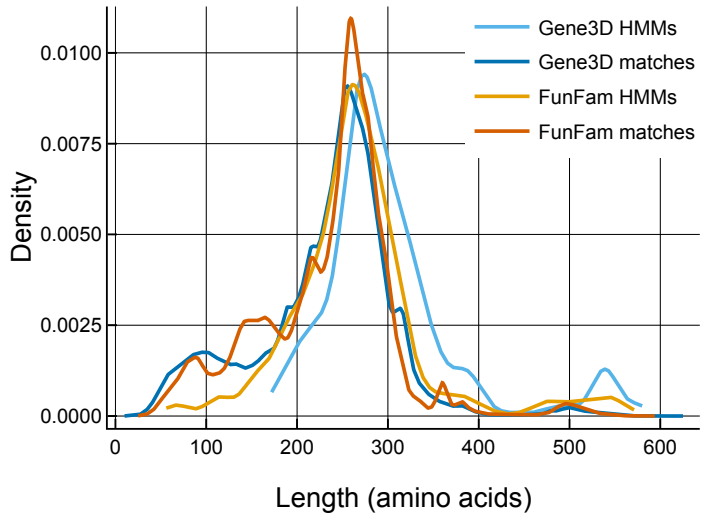


Figure 3.8: **Length distribution of ABH superfamily and ABH FunFam matches.** ABH HMMs from Gene3D and FunFams are also plotted. The probability density function of each distribution was estimated using kernel density estimation.

3.3.3 α/β hydrolase domains are enriched in non-natural biomes

We examined whether any biomes are enriched with ABH domains and may be promising biomes to search for candidate plastic-degrading enzymes. A linear relationship exists between the number of sequences found in a biome and the number of ABH domains found in those sequences (Fig. 3.9). This means that ABH domains occur at a constant rate in nature. Most biomes contain the expected number of ABH domains, given the number of proteins that were found in the biome. According to a fitted regression model, 'Engineered' biomes have significantly more ABH domains than expected (Fisher's $P \approx 0$). Engineered biomes encompass a broad range of non-natural biomes, including industrial settings, laboratory conditions or waste treatment. The regression model also predicts that proteins from 'Human' and human 'Digestive system' biomes are depleted with ABH domains (Fisher's $P \approx 0$).

3.3.4 α/β hydrolase domains in metagenomes are diverse

To understand the diversity and novelty of ABH domains in metagenomes, we clustered the 1,065,976 ABH domain sequences from MGnify (508,693) and UniProt (557,283) at 30%, 50%, 70% and 90% sequence identity. As the clustering sequence identity threshold is increased, the number of clusters increases, producing a large number of clusters at high sequence identity (Fig. 3.10). At S90, there are 755,547 clusters, which shows that

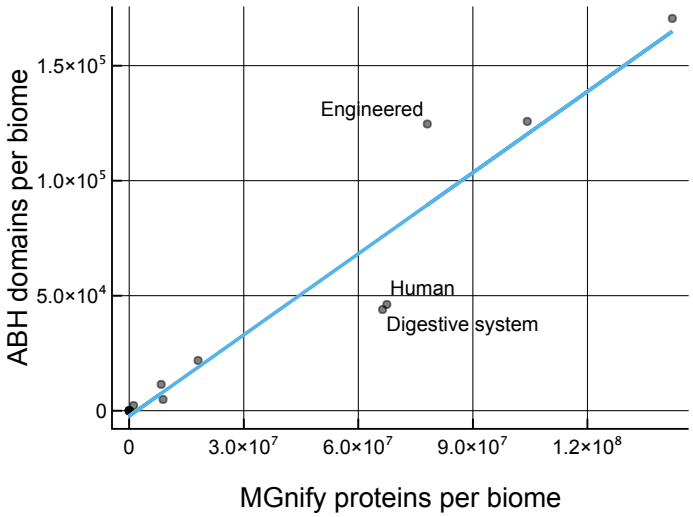


Figure 3.9: **Relationship between the number of MGnify proteins and the number of ABH domains per biome.** A linear regression model was fitted to the data and plotted. Biomes that deviate from the regression line are labelled.

319,196 sequences (30%) share more than 90% sequence identity with another sequence in the data set. As the MGnify proteins are S90 cluster representatives, they will not cluster together at S90. Therefore, some UniProt and MGnify sequences may be clustersing into mixed origin clusters (Section 3.3.5).

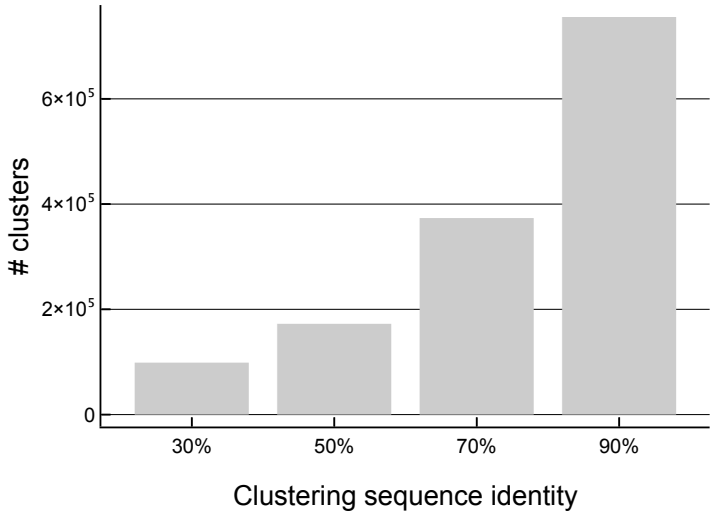


Figure 3.10: **Number of S90 clusters in MGnify and UniProt ABH domains.** 1,065,976 ABH domain sequences from MGnify (508,693) and UniProt (557,283) at 30%, 50%, 70% and 90% sequence identity using Linclust.

As the sequence identity threshold increases, the number of singleton clusters, that

contain a single sequence, increases rapidly (Fig. 3.11). 230,666 ABH domain sequences (21%) share less than 70% sequence identity with all other sequences and are singletons. These singletons could represent novel functions, whose sequence diversity is not represented in gold standard databases, such as UniProt. Conventional wisdom states that protein function is conserved to approximately 60% sequence identity. An analysis in 2002 found that $< 30\%$ of proteins with $> 50\%$ sequence identity have exactly the same function, according to all four digits of the EC annotation being the same [8]. But hard and fast sequence identity thresholds of functional conservation for enzymes are unwise because catalysis and substrate-specificity are often determined by only a small number of residues [398].

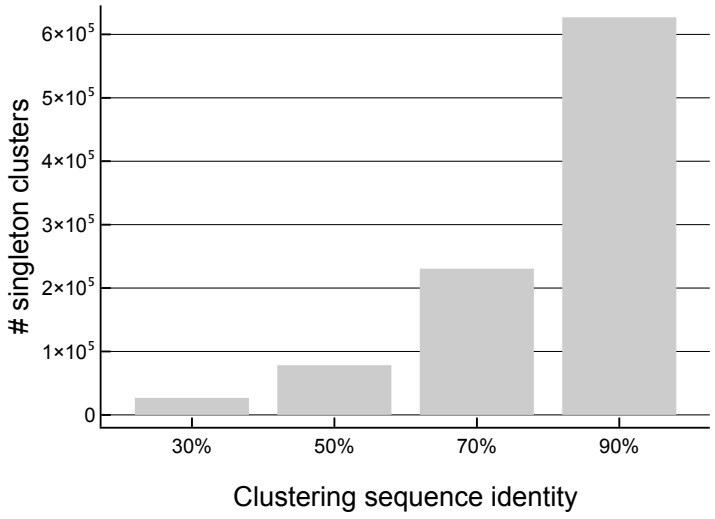


Figure 3.11: **Number of singleton S90 clusters in MGnify and UniProt ABH domains.** 1,065,976 ABH domain sequences from MGnify (508,693) and UniProt (557,283) at 30%, 50%, 70% and 90% sequence identity using Linclust.

3.3.5 α/β hydrolases in metagenomes are more similar to prokaryotic, than eukaryotic, ABH domains in UniProt

We assessed whether ABH domains from MGnify proteins are more similar to UniProt ABH domains from prokaryotes or eukaryotes. Shotgun metagenomics aims to identify all DNA from a biome. Whilst much of this genetic material will originate from prokaryotes, a significant fraction will be from eukaryotes and viruses. The question is: Are ABH domains from MGnify proteins most similar to prokaryotic or eukaryotic ABH domains in UniProt?

To answer this question, we considered mixed clusters composed of ABH domains from MGnify and UniProt. UniProt is taxonomically biased, with a prokaryotic-to-

eukaryotic sequence ratio of $3:1$ (75%) (Fig. 3.12 UniProt all). For proteins containing ABH domains, UniProt remains biased towards prokaryotes (64%, Fisher's $P \approx 0$), but is less biased than all of UniProt (Fig. 3.12 UniProt ABH). Mixed clusters follow a Bernoulli distribution that models the probability that a UniProt sequence is prokaryotic. The null hypothesis for mixed clusters is $B(0.64)$.

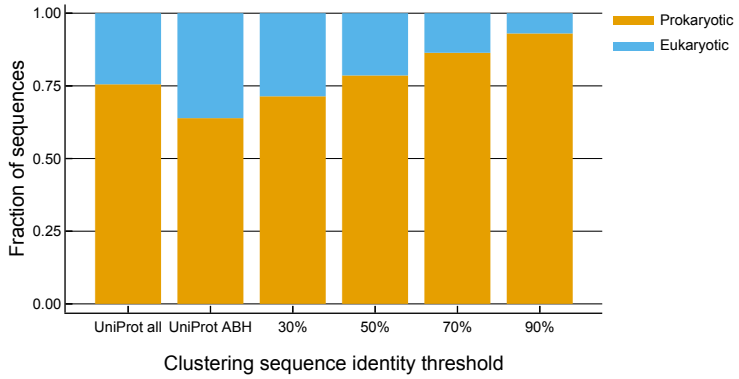


Figure 3.12: **Assessing the similarity of MGnify ABH domains to UniProt ABH domains from prokaryotes or eukaryotes.** $1{,}065{,}976$ ABH domain sequences from MGnify ($508{,}693$) and UniProt ($557{,}283$) at 30%, 50%, 70% and 90% sequence identity using Linclust. Prokaryotic-to-eukaryotic ratios are plotted at each sequence identity threshold for UniProt sequences in mixed clusters. The prokaryotic-to-eukaryotic ratio for all UniProt sequences (UniProt all) and ABH domains in UniProt (UniProt ABH) is also plotted.

ABH domains from MGnify proteins cluster with ABH domains from UniProt, showing that metagenomes contain, at least some, previously identified sequence diversity contained in UniProt. The prokaryotic fraction of these mixed clusters increases at higher sequence identities (Fig. 3.12) (Fisher's $P \approx 0$ for testing all sequence identity threshold clusterings against UniProt all or UniProt ABH). There may be a number of causes for this effect, including the high fraction of non-culturable species in metagenomes, how microbiome samples are prepared before sequencing and how the protein sequences were predicted in the microbiome assemblies. These points are discussed further in (Section 3.4.3).

We next examined the taxonomy of UniProt sequences in mixed clusters at 30%, 50%, 70% and 90% sequence identity. Whilst the number of mixed clusters remains stable across the sequence identity thresholds (data not shown), the prokaryotic fraction increases at higher sequence identities (Fig. 3.12).

Many metagenomic ABHs are novel and rare, but those that are common in MGnify are also found in UniProt (Fig. 3.13). Most ABHs from MGnify are rare and found in small

clusters with fewer than 10 sequences. Many of the rare ABHs are also functionally novel because they are not represented in UniProt. Only 20% are in mixed clusters with UniProt ABHs. On the contrary, functions of common metagenomic ABHs are already represented in UniProt. 85% of clusters with 10 or more sequences are mixed.

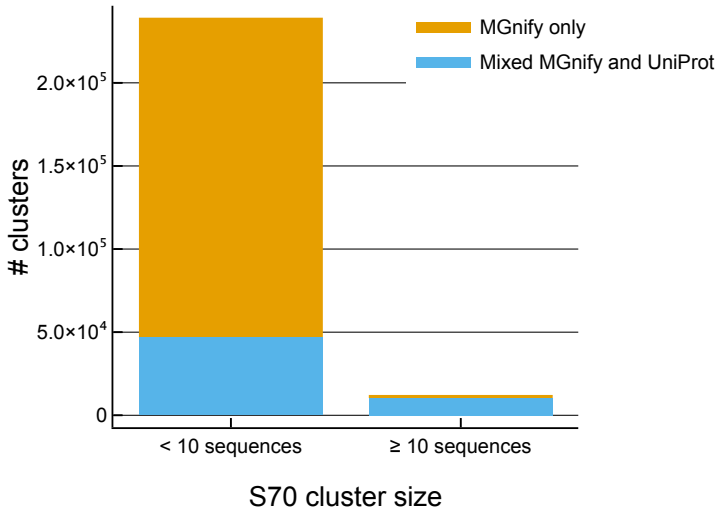


Figure 3.13: **Analysis of rare and common MGnify ABH domains.** 1,065,976 ABH domain sequences from MGnify (508,693) and UniProt (557,283) at 70% sequence identity using Linclust. Clusters containing MGnify sequences were grouped by size into clusters with < 10 sequences or ≥ 10 sequences. The number of clusters is plotted grouped by whether the cluster contains 'MGnify only' ABHs, or 'mixed MGnify and UniProt ABHs'.

3.3.6 Little evidence for evolution of novel domains in metagenomes

The MGnify protein sequences used in this study were sampled from biomes—and species—that have, presumably, been largely unexplored. Species in these biomes may have evolved into regions of sequence space that laboratory strains, model organisms and other well-studied species have not. These regions of sequence space may encode novel folds, domains or functions. So, have new domains evolved in metagenomes?

To answer this question, we analysed terminal regions and inter-domain sequences, i.e. contiguous regions of sequence that do not have significant matches to any Gene3D HMMs in MDAs (Section 3.2.7). These regions could be novel domains that have evolved in metagenomes and are, as yet, unknown. For conceptual simplicity, we considered full-length proteins whose MDAs only contain continuous domains. Therefore, single-domain proteins only have terminal sequences at each termini, whereas, multi-domain proteins can also have one inter-domain sequence between each pair of adjacent domains (Sec-

tion 3.2.7).

The distribution of inter-domain sequence lengths is shown in (Fig. 3.14). The modal value is a gap length of 0 residues, i.e. contiguous domains. Gap length probability decreases exponentially with a median of 3 residues and a mean of 22 residues. For comparison, CATH S95 model lengths are also plotted, which follow a positively skewed normal distribution, or log-normal distribution. The median length of these models is 145 residues, yet 95.4% of the gaps in the MGnify protein sequences are less than 145 residues long. Furthermore, 64.8% of the gaps are shorter than the shortest HMM, which is 16 residues long.

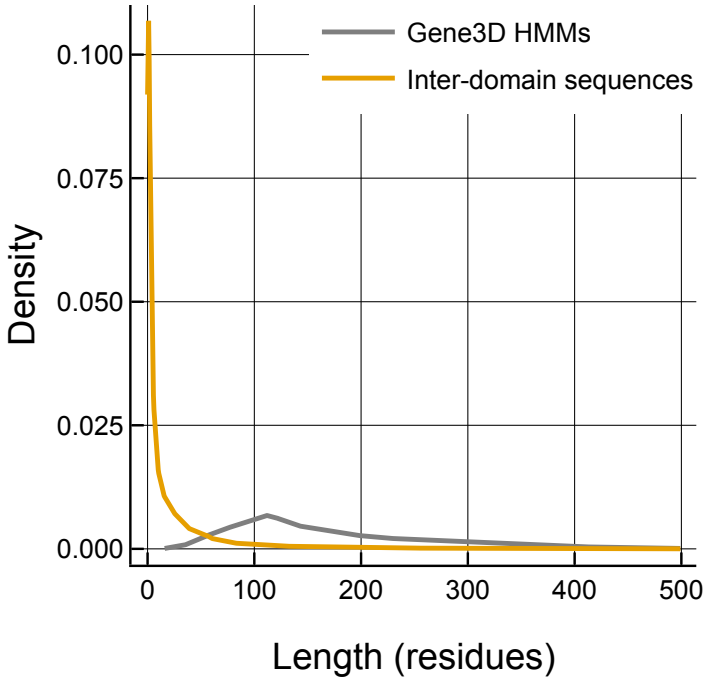


Figure 3.14: **Inter-domain sequence lengths.** The distribution of inter-domain sequences lengths that are not contained in MDAs is plotted. For comparison, the distribution of Gene3D HMM lengths is shown. The x-axis is truncated at 500 residues. The probability density function of each distribution was estimated using kernel density estimation.

As such, terminal and inter-domain regions in metagenome proteins containing an ABH domain are unlikely to be novel domains. In order to confirm this, these sequence regions could be scanned against Pfam or PDB.

3.3.7 Identification of errors in protein sequence clustering

We noticed that some ABH domain sequences found in the MGnify proteins were not unique. Please note that the MGnify protein data set that we used in this analysis were cluster representatives. For identical domain sequences to be in different S90 clusters, the following is true: regions of sequence that flank identical ABH domains must be sufficiently different to reduce the overall sequence identity below 90%. We investigated these flanking regions to determine how similar the overall sequence is for proteins that contain identical ABH domains. For sequences that contain identical ABH domains, we pairwise aligned the full-length sequences using global sequence alignment with a constant gap penalty. Most pairs of sequences that have identical ABH domains are very similar across the entire sequence length (median distance $= 0.011$; Fig. 3.15). What's more, 61 sequence pairs have a distance of zero. Not only are the domain sequences identical, but the entire sequences are completely identical. Finally, few sequence pairs have a distance > 0.1, which is approximately equal to sequence identity $< 90\%$.

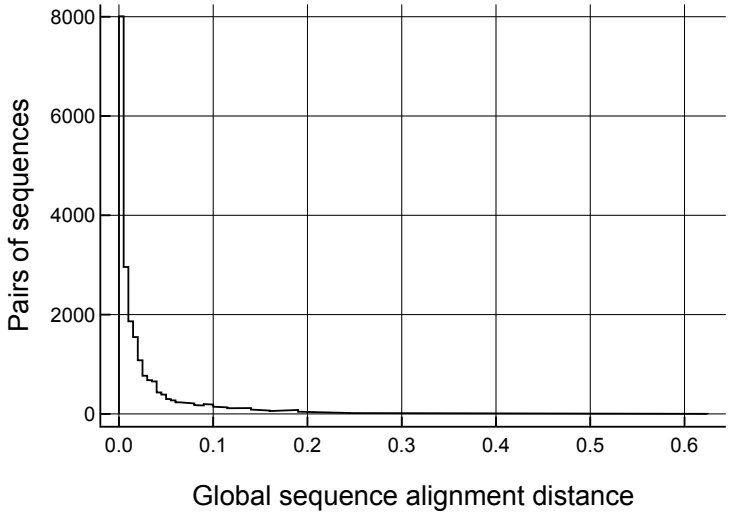


Figure 3.15: **Sequence alignment distances for pairs of MGnify proteins that contain identical ABH domain sequences.** Full-length sequences were globally aligned with a constant gap penalty. Alignment scores were converted to distances using, $D_{S_1S_2} = 1 - \frac{A_{S_1S_2}}{\min(A_{S_1S_1}, A_{S_2S_2})}$, where A is the global alignment score and D is the alignment distance between two sequences S_1 and S_2.

Whilst errors in clustering are not ideal, only $3{,}676$ unique domains in $9{,}379$ sequences are affected. We should not worry too much about this too much, as $9{,}379$ sequences is less than 1% of the $508{,}693$ sequences that contain an ABH domain. But this

conclusion is not satisfactory to a researcher. Why were these identical, or nearly identical, sequences not clustered together? Put simply, Linclust trades off accuracy for speed. The MMseqs2 issue tracker on GitHub (https://github.com/soedinglab/MMseqs2) has two issues, #88 and #104, related to identical sequences not being clustered together. The solution suggested by the developers is to increase the number of k-mers selected from each sequence from 21 to 80. Doing so will, on average, increase the number of k-mers that are shared between sequences and cluster centroid sequences, which will increase the probability of identical sequences ending up in the same cluster. The runtime memory requirements will quadruple from 400 GB to 1.6 TB. Given the current database size, this would be feasible on EBI's 1.9 TB 'big memory' machines. The sequence database need only grow by 25% before this approach would no longer be tenable. To counter exploding memory requirements, MMseqs2 provides an option to load chunks of sequences into memory, at the expense of speed. We have not tested these approaches yet and debugging this workflow is tedious because it takes $\sim$ 24 hours to run. But, in time, such solutions will need to be explored as the database size increases.

3.3.8 α/β hydrolase domains are distributed amongst FunFams differently in MGnify and UniProt

We explored the distribution of ABH FunFam domains in the MGnify proteins and compared it to the distribution in UniProt. CATH v4.2 has 377 ABH FunFams. ABH domains from the MGnify proteins match 148 of the 377 ABH FunFams (39%; Fig. 3.16) using the per FunFam inclusion thesholds. Domains are not distributed in these 148 FunFams the same as ABH domains in UniProt. For example, the largest FunFam in UniProt (3.40.50.1820/FF/115309) has 53,144 members (16% of ABH domains in the 148 FunFams), but is 70$^{\text{th}}$ largest in MGnify with only 34 members ($\sim$ 0%). This FunFam is associated with one molecular function: 'Hydrolase activity, acting on ester bonds' (GO:0016788). Conversely, the largest FunFam in MGnify (3.40.50.1820/FF/115552) has 131,349 members (37%) and is the fourth largest in CATH with 20,873 members (6%). This FunFam is also associated with GO:0016788, as well as many other molecular function and biological process annotations. These include molecular functions 'Chlorophyllase activity' (GO:0047746), 'Pheophytinase activity' (GO:0080124) and 'Bromide peroxidase activity' (GO:0019806); and biological processes 'Chlorophyll metabolic process' (GO:0015994), 'Response to toxic substance' (GO:0009636) and 'Aromatic compound catabolic process'

(GO:0019439).

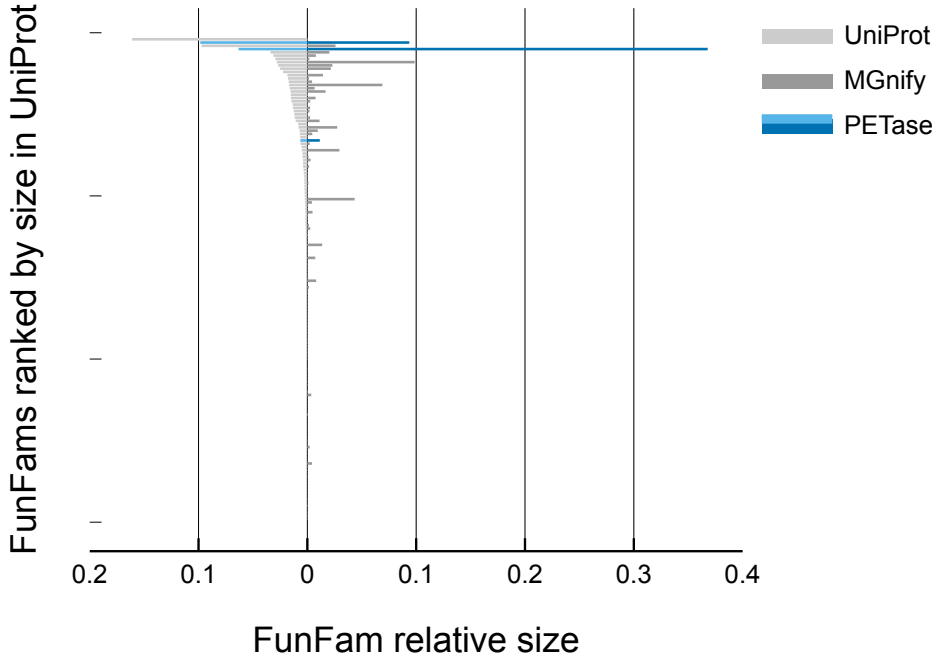


Figure 3.16: **Distribution of ABH FunFams in MGnify and UniProt.** Of the 377 ABH FunFams in CATH v4.2, ABH domains in the MGnify proteins match to 148. These 148 common FunFams are ranked by their size in UniProt. Relative size is calculated by normalising each FunFam by the total number of matches in UniProt and MGnify respectively, so that the bars on each side of the y axis sum to one. FunFams that PETase matches to using the FunFam inclusion threshold are highlighted in blue.

PETase matches to three ABH FunFams (3.40.50.1820/FF/115552, 3.40.50.1820/FF/115534 and 3.40.50.1820/FF/115660) that are associated with functions that are similar to hydrolytic depolymerisation of PET. The largest FunFam for the MGnify proteins, 3.40.50.1820/FF/115552, was discussed in the previous paragraph. PET is a polymer of ethylene terephthalate, an aromatic monomer, which is similar to large, complex organic molecules, such as chrlorophyll, pheophytin and steroid hormones. 37% of MGnify ABH domains are in this FunFam. The second largest FunFam, 3.40.50.1820/FF/115534, contains 9% of MGnify ABH domains, which corresponds to the 10% of UniProt ABH domains in this family. This FunFam is associated with 19 molecular function, and 32 biological process, GO terms related to lipid hydrolysis, signalling and inflammatory responses. The smallest FunFam, 3.40.50.1820/FF/115660, is proportionally 1.8 times larger than in UniProt. This FunFam is associated with 19 molecular function, and 24 biological process, GO terms related to lipid hydrolysis and steroid hormone signalling. Chemically and

structurally, PET resembles lipids, which are polymers of organic monomers.

3.3.9 Generating FunFams at gigascale

- Harry Scholes designed the FRAN and $\text{FRAN}_{\text{geometric}}$ algorithms.
- Clemens Rauer implemented the algorithms and ran the benchmarks.
- Harry Scholes analysed and interpreted the benchmarking results.

Sequence databases are growing rapidly in size and diversity, which, in turn, benefits the quality of FunFams. Can our methods cope with billions of sequences at gigascale? Here, we benchmark two new algorithms, FRAN and $\text{FRAN}_{\text{geometric}}$, to generate FunFams on arbitrarily large numbers of sequences (Section 3.2.10).

We tested the performance of FRAN and $\text{FRAN}_{\text{geometric}}$, considering the quality of the resulting FunFams. For comparison, we benchmarked against GARDENER, the gold standard method to generate FunFams. We tested these three approaches on α/β hydrolases (Table 3.1). All three methods generate a similar number of FunFams, with a similar number of FunFams having EC terms. FRAN (1048) and $\text{FRAN}_{\text{geometric}}$ (1042) generate very similar numbers of FunFams. Clusterings are very similar to GARDENER, as measured by the Rand index, for FRAN (0.87) and $\text{FRAN}_{\text{geometric}}$ (0.88). FRAN and $\text{FRAN}_{\text{geometric}}$ form almost identical clusters, as measured by the Rand index of 0.95.

Table 3.1: **FunFams generated by FRAN, $\text{FRAN}_{\text{geometric}}$ and GARDENER.** FunFams were generated for the α/β hydrolase family, CATH superfamily 3.40.50.1820. The number of starting clusters, FunFams, FunFams with EC terms and the Rand index of FunFam agreement with GARDENER are reported.

Method	Starting clusters	FunFams	FunFams with EC	Rand index
GARDENER	1546	1188	298	1.00
FRAN	1546	1048	255	0.87
$\text{FRAN}_{\text{geometric}}$	1546	1042	274	0.88

We confirmed that FRAN and $\text{FRAN}_{\text{geometric}}$ parition starting clusters into similar FunFams by constructing graphs of FunFams and calculating graph theoretic measures on them (Fig. 3.17). FRAN, $\text{FRAN}_{\text{geometric}}$ and GARDENER have as many maximal cliques as connected components because FunFams are hard clusters. FunFam clustering by FRAN and $\text{FRAN}_{\text{geometric}}$ have very similar global agreement, as shown by the graph union of GARDENER and FRAN (G ∪ F) having approximately the same number of connected components and maximal cliques as the graph union of GARDENER and $\text{FRAN}_{\text{geometric}}$ (G ∪ Fg). Both graph unions are below the $y = x$ line, which shows that they have fewer

connected components than maximal cliques. This means that FRAN and $\text{FRAN}_{\text{geometric}}$ generate different FunFams to GARDENER, but that these FunFams form larger connected components, comprised of multiple maximal cliques, in the graph union because some starting clusters are shared between the FunFams. This is confirmed by the graph union of FRAN and $\text{FRAN}_{\text{geometric}}$ (F $\cup$ Fg), which has fewer connected components than maximal cliques, showing that many of the FunFams generated by these two methods intersect. Finally, the graph union of all three methods (G $\cup$ F $\cup$ Fg) has slightly fewer cliques than G $\cup$ F, or G $\cup$ Fg, but far fewer connected components. More FunFams are being merged into the same connected components in (G $\cup$ F $\cup$ Fg) because FRAN and $\text{FRAN}_{\text{geometric}}$ parition the starting clusters into different FunFams. However, these FunFams intersect with other FunFams generated by the other method and by GARDENER. Whilst FunFams generated by FRAN and $\text{FRAN}_{\text{geometric}}$ do not agree perfectly with GARDENER, the FunFams are similar, as shown by the large number of FunFams that have intersecting starting clusters. Overall, the FRAN and $\text{FRAN}_{\text{geometric}}$ algorithms produce sufficiently similar FunFams to be taken forward for further analysis.

The EC purity of FunFams generated by FRAN and $\text{FRAN}_{\text{geometric}}$ were similar to GARDENER (Fig. 3.18). There was no significant difference between the EC purity distribution for FunFams produced by GARDENER and FRAN (two-sided Mann-Whitney $P = 0.72$), or GARDENER and $\text{FRAN}_{\text{geometric}}$ (two-sided Mann-Whitney $P = 0.85$). Therefore, the functional purity of FunFams generated by FRAN and $\text{FRAN}_{\text{geometric}}$ were comparable to FunFams generated by GARDENER.

As FRAN and $\text{FRAN}_{\text{geometric}}$ produce very similar FunFams, but $\text{FRAN}_{\text{geometric}}$ is more computationally expensive, FRAN will be taken forward to be used to generate FunFams of large CATH superfamilies.

3.4 Discussion

3.4.1 A large number of α/β hydrolase domains were found in metagenomes

Whilst MGnify contains many analyses of metagenomic sequencing studies (315,181 as of April 14, 2020), only a small subset (6%) of studies have so far been assembled (18,291 as of April 14, 2020). The number of protein sequences identified in each biome probably does not reflect the sequence and functional diversity of the biome (Fig. 3.5). Rather, the number of sequences reflects the degree to which each biome has been sampled, which samples have been assembled, or which assemblies have had ORFs predicted. For example,

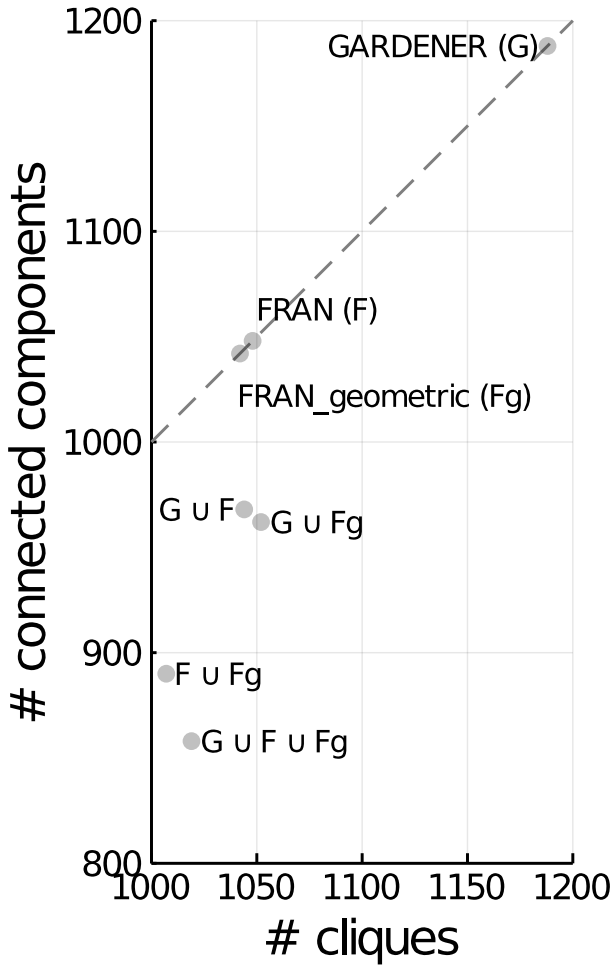


Figure 3.17: **FunFam graph theoretic benchmarks for FRAN (F), FRAN$_{\text{geometric}}$ (Fg) and GARDENER (G).** Graphs were constructed for FunFams that are composed of > 1 starting cluster. Perfect FunFam clustering is shown as a dotted line at $y = x$. $\cup$ denotes the graph union operation.

there have been vast metagenomics projects of marine [301] and human gut [299, 300] microbiomes that will have captured the sequence diversity in these biomes. Comparatively, plants, soil and other animal biomes have been neglected.

The modified protocol that we used to find superfamily domains in large sequence data sets (Section 3.2.4.2) saved vast computational resources. Scanning 10^6 sequences MGnify proteins against the 416 ABH Gene3D HMMs took approximately 30 minutes on four cores—that is, 3.6 CPU weeks in total. Scanning 10^4 sequences against all Gene3D HMMs took approximately 60 minutes on four cores—that is, 3.9 CPU weeks for all 1,630,166 sequences that had a significant ABH hit. Using these timings, we can esti-

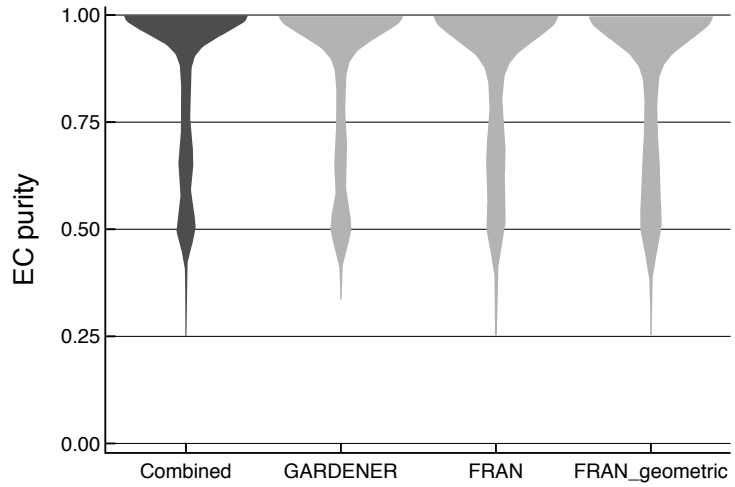


Figure 3.18: **FunFam EC purity benchmarks for FRAN, FRAN$_{\text{geometric}}$ and GARDENER.**

mate that searching all of the MGnify proteins against all Gene3D HMMs would take 14 CPU years, whilst searching the redundant database of 1.1 billion sequences would take 50 CPU years. Additionally, by implementing this pipeline in Nextflow, we could conveniently split up the sequence data sets into managable chunks, that could be processed in an embarrassingly parallel way, by many independent jobs, each with low memory and CPU requirements.

We identified a large number of diverse ABH domain sequences from full-length and truncated ORFs (Fig. 3.6). It is reasonable to assume that sequences truncated at one terminus would, on average, be longer than sequences truncated at both termini. However, the median length of sequences truncated at either terminus is 136 residues, whereas, sequences truncated at both termini have median length 164 residues. This could have been caused by the Prodigal algorithm [339]. Prodigal looks for ribosome binding sequence motifs, such as the Shine-Dalgarno sequence, and an in-frame stop codon to predict proteins from ORFs. To reduce the number of false positive ORFs, Prodigal penalises sequences shorter than 250 bp by a linear factor that is proportional to their length [339]. These features are strong, so short sequences that are truncated at one end can still end up with favourable scores. However, sequences that are truncated at both ends must be long to have favourable scores.

Because we identified a large number of ABH domains, we took full-length sequences, and any ABH domains contained in full-length sequences, forward for further analysis.

ABH domains identified in the MGnify proteins follow the same length distribution as known ABH domains from UniProt (Fig. 3.7). This finding indicates the quality of MGnify data. Firstly, it suggests that the MGnify assemblies represent real contigs found in microbiomes. Secondly, and *propter hoc*, the predicted ORFs and protein sequences real.

70% of ABH domains in MGnify, predicted by Gene3D, were assigned to an existing ABH FunFam using the strict per FunFam inclusion threshold (Fig. 3.8). There are a number of possible causes for this. Firstly, new functions could have evolved in metagenomes. These functions would not be present in UniProt. Because FunFams are generated using sequences from UniProt, FunFams will not be generated that represent these newly-evolved functions. Secondly, FunFams are generated from starting clusters that are associated with an experimentally characterised function. ABHs with novel and uncharacterised functions will not be covered by FunFams.

It is encouraging that ABH domains were found in engineered biomes more frequently than the background rate for MGnify proteins (Fig. 3.9). This finding may represent an increased potential to find proteins with biotechnology applications similar to PETase, which was found in an engineered environment [374]. Conversely, ABH domains were depleted in the human digestive system (Fig. 3.9). Whilst α/β hydrolase proteins are present in the digestive system to degrade proteins and lipids, the diversity of these proteins may not be that high. Considering that the MGnify proteins are cluster representatives, the importance (concentration) of these enzymes may be obscured by the lack of sequence diversity.

3.4.2 α/β hydrolase domains in metagenomes are diverse

To understand their diversity and novelty, ABH domains from MGnify proteins were clustered with ABH domains from UniProt proteins. A large number of clusters (Fig. 3.10) and singleton clusters (Fig. 3.11) result at high sequence identity. This demonstrates that the ABH domain fold can accommodate a high diversity of sequences, whilst leaving the structure intact. Evidently, this robustness has been exploited by evolution to produce a wide variety of ABH domain functions.

Sequences clustered together, as shown by the number of clusters always being much lower than the number of sequences. The MGnify proteins are S90 cluster representatives. So this result suggests that some of the MGnify ABH domain sequences are clustering with those from UniProt. It should be noted that the domain sequences from UniProt have not

been filtered to remove redundant sequences, however, it is unlikely that only the UniProt sequences clustered together. If MGnify and UniProt ABH domains clustered together, this would further increase our confidence in the quality of the assemblies and ORFs in MGnify.

It is remarkable that 21% of ABH domain sequences are singletons at S70. This means that, whilst the overall chemistry of the catalysis may be the same, these sequences are likely to have different (specific) functions, or be regulated differently. Additionally, the large number of singletons at S70 suggests that biomes have not been sampled exhaustively. Many more functions are out there waiting to be discovered. As more biomes are studied and more samples are collected, I expect the number of clusters and singletons to continue to increase.

Given both of these findings, it appears that clustering at S70 or S90 is too stringent for the degree of sequence diversity in a superfamily and clustering at S60 is more likely to obtain meaningful clusters. Clustering at this level makes sense because protein function is conserved to approximately 60% sequence identity [8, 398]. Whilst this sequence identity threshold may be true for ABH domains, it may not be true for the other CATH superfamilies.

3.4.3 α/β hydrolases in metagenomes are more similar to prokaryotic, than eukaryotic, ABH domains in UniProt

UniProt is biased towards prokaryotes, but this bias is less so in proteins that contain ABH domains (Fig. 3.12). We used the taxonomic bias for proteins containing ABH domains to define a null distribution for mixed clusters. Other biases may affect the taxonomic distribution of proteins in MGnify, for example which biomes were sampled, how high the coverage of sampling was, how samples were prepared prior to sequencing, which samples were assembled, and choice of gene callers. Despite these biases, we can arguably be more confident about mixed, rather than single origin, clusters because these sequences are present (independently) in both MGnify and UniProt.

We observed that the prokaryotic fraction of mixed clusters increases at higher sequence identities (Fig. 3.12). There may be a number of causes for this effect. Firstly, UniProt may not contain representatives of eukaryotic ABH domains found in metagenomes. Metagenomes contain a large number of unknown species that cannot yet be cultured. It has been estimated that only 1% of microorganisms have been cultured

[295]. Many of these species are likely to be eukaryotic [297], whether small or single-celled. It is reasonable to assume that ABH domains have evolved in these species to allow them to occupy particular niches. Therefore, it is likely that these domain sequences will not be present in UniProt. Secondly, metagenomic samples are size fractionated to remove larger objects in samples, so eukaryotes are more likely to be removed and prokaryotes are more likely to be retained. Thirdly, MGnify only uses prokaryotic gene callers to predict ORFs in assemblies (Section 3.2.1.1). Prokaryotic gene callers may have a high false negative rate on eukaryotic contigs. Eukaryotic gene callers, such as GeneMark-EP [399], MetaEuk [400], or EuGene [401], could be applied in parallel with the current prokaryotic gene callers. These three factors will contribute to eukaryotic proteins being underrepresented in the MGnify proteins. This analysis should be repeated when MGnify incorporates eukaryotic gene calling into its protein prediction pipeline.

Overall, presence of mixed clusters shows that ABH domains in metagenomes are not distinct from previously known sequences in UniProt. Rather, we see sequence conservation alongside evolution of new functions. New functions are generated by exploration of previously unexplored regions of sequence space to enable organisms to survive and occupy different niches. Retention of function by an organism is a fitness cost: if functions are not required, they will be lost. Conservation of previously-known functions demonstrates that these functions are required by species to survive in particular biomes

3.4.4 Little evidence for the evolution of novel domains in metagenomes

We assessed whether novel domains had evolved in metagenomes (Fig. 3.14), but found little evidence for it. Inter-domain lengths were short, so the vast majority of metagenomic protein sequence is covered by a significant Gene3D hit to a CATH superfamily. Therefore, it appears that novel domains have not evolved in metagenomes. There are, however, some caveats to this conclusion. Although we have little evidence of novel domain evolution in proteins containing ABH domains, we cannot extrapolate these conclusions to metagenomic protein sequences in general. Similar analyses on different superfamilies, subsets of superfamilies, or all superfamilies would be required before drawing general conclusions about metagenomes. Further caution should be exercised because 91% of sequences containing ABH domains in Pfam are single-domain proteins (73,297 out of 80,360 sequences in Pfam family PF00561), which only have two terminal regions and no inter-domain regions.

3.4.5 α/β hydrolase domains are distributed amongst FunFams differently in MGnify and UniProt

As FunFams are functionally pure, functions of proteins can be predicted by mapping domains to FunFams. The ABH domain from PETase was mapped to three FunFams involved in the hydrolysis of large organic biomolecules (Fig. 3.16). PET is a polymer of ethylene terephthalate, a monomer composed of an aromatic ring with carboxy groups at the 1 and 4 ring positions. A carboxy group at the 1 position of one monomer reacts with a second monomer's ethanol moiety attached to the single-bonded oxygen of the carboxy group at the 4 position. PET resembles the substrates of proteins that map to the same FunFams. Lipids are carbon polymers. Many plastics, including PET, are polyesters, i.e. polymers formed by esterification between carboxylic acids and alcohols of monomers. Chrlorophyll, pheophytin and steroid hormones are composed of many aromatic groups. These findings naturally give rise to a hypothesis for the origins of PETase. A hydrolase, whose cognate ligand is some type of large organic biomolecule, evolved to degrade PET under massive selection pressures.

In addition, MGnify ABH domains were mapped to FunFams (Fig. 3.16). One possible reason for finding so few matches in MGnify to the largest ABH FunFam in UniProt (3.40.50.1820/FF/115309) could be that the FunFam is large and so the sequence alignment might be poor. 16% of UniProt ABH domains are in this family. Thus, the resulting HMM would not be very informative, so few sequences would have high-scoring matches. Low-scoring matches would be trumped by higher-scoring matches when the MDAs were resolved. 3.40.50.1820/FF/115552 contains 37% of MGnify ABH domains. Compared with UniProt, this family is significantly expanded in metagenomes, which suggests that this FunFam increases the fitness on species living in diverse microbiomes. This family is associated with hydrolysis of ester bonds and large biomolecules. This might mean that hydrolases have evolved to metabolise other synthetic man-made materials, such as other types of plastics. These materials are certainly in the environment, so there is a selection pressure for organisms to make use of these energy sources. If so, it is likely that the ABH domain will evolve to degrade these materials because of its incredible functional plasticity.

3.4.6 The FRAN algorithm allows FunFams to be generated at gigascale

Presently, we are facing challenges generating FunFams of large superfamilies using GARDENER because all starting clusters are required to be kept in memory as the GeMMA tree is grown. Our current workaround is to run large superfamilies on a high-memory machine with 3 TB memory, but we are already approaching the memory limit of these machines. Memory is expensive and 3 TB is already a lot of memory, so we need to develop a low-memory strategy that will allow FunFams to be scaled to extremely large superfamilies in the future. Whole-genome sequencing and metagenomics are now possible, are being adopted and are growing in popularity. As such, a scalable protocol would allow FunFams to be generated using protein sequences from metagenomes and from UniProt. Also, we currently only generate FunFams from S90 clusters that have at least one experimental GO term annotation. A scalable protocol will allow us to remove this restriction.

The current restriction means that every FunFam is associated with a known function, which is useful for function prediction. However, this means that FunFams cannot be generated for proteins with novel functions. If a FunFam has no annotated functions, putative functions could be predicted by transferring any functions from the nearest k neighbouring FunFams, within some E-value radius r. Neighbouring FunFams can be found by pairwise HMM alignment to all other FunFams in the same superfamily. These putative functions will be useful to experimentalists to guide their choice of proteins to validate, which will be particularly important when validating proteins with novel functions from metagenomes. Alternatively, FunFams can be annotated *post hoc*, whenever one of the members has been experimentally characterised. We hypothesise that the number of FunFams will increase dramatically to reflect the increase in sequence and functional diversity present in these additional sequences.

Here, we designed two divide-and-conquer algorithms, FRAN and $\text{FRAN}_{\text{geometric}}$, to generate FunFams on subsets of a superfamily's sequences at a time. The algorithms first sample starting clusters into different groups, which are used as independent inputs to GeMMA and FunFHMMer. The FunFHMMer outputs are then pooled and treated as starting clusters for a second round of GeMMA and FunFHMMer, which allows sequences that were sampled into different groups to be merged into the same FunFams. Curiously, we observed no difference between the quality of FunFams that FRAN, $\text{FRAN}_{\text{geometric}}$ and GARDENER generated (Figs. 3.17 and 3.18).

Originally, we hypothesised that if a superfamily's sequences were subset into different groups, each group should—as far as possible—retain the same characteristics as all of the sequences combined. Comparing the performance of FRAN to $\text{FRAN}_{\text{geometric}}$ shows that uniform random sampling produces as good FunFams to the geometric sampling strategy, employed to respect the evolutionary relationships between sequences. This result demonstrates the power of GARDENER at being able to fix any clustering errors in the first round of FunFamming with a second round.

$\text{FRAN}_{\text{geometric}}$ is more expensive than the null model because it requires two clustering steps at S90 and S30, as well as drawing two random numbers when sampling each starting cluster into groups—one random number to sample an S30 and one random number to sample an S90 contained therein. Conversely, FRAN requires one clustering step at S90, followed by one random number to sample each starting cluster. Going forward, FRAN will need to be benchmarked on more superfamilies and compared to GARDENER. If the performance is comparable across many superfamilies, then FRAN should be used to generate FunFams of problematically large superfamilies, whilst vanilla GARDENER could be used on manageable superfamilies.

3.4.7 Conclusion

This work laid the foundations for two exciting new avenues of research. First, we conducted a proof-of-concept study into mining very large protein sequence data sets for novel functions. We used our tools—CATH, Gene3D and FunFams—to search metagenomes for plastic-degrading enzymes similar to PETase. Sequence databases are continue to grow in size, so following on from this, we investigated approaches that would allow FunFams to be generated on an arbitrarily large number of sequences. We decomposed the FunFam generation algorithm, GARDENER, using a divide-and-conquer random sampling approach, FRAN. This approach will allow FunFams to be generated on ever-growing sequence databases, including metagenomes, future-proofing FunFams for many years to come.

Chapter 4

Feature learning from graphs using unsupervised neural networks

4.1 Introduction

In this chapter, we explore a state-of-the-art protein function prediction method, deep network fusion (deepNF), that uses protein networks as its sole training data.

Graphs are ubiquitous data structures that are suited to modelling problems that are high-dimensional and sparse. As such, graphs are an ideal data structure to represent the myriad protein interactions that give rise to biological life. However, due to many biological and experimental factors, networks are models that try to capture as many of the true interactions that proteins make, but are often incomplete.

Given some protein network, one may want to perform link prediction to predict additional interactions that proteins might make, but are missing from the network. In the past decade, machine learning has become the de facto framework to model prediction problems. Due to memory constraints, it may not be practically feasible to represent graphs as dense matrices of features, as is required for many types of machine learning algorithms. For example, storing a dense adjacency matrix of a 10^6 node graph in 64-bit precision requires 80 GB of memory. Furthermore, even if dense graph matrices can be stored, machine learning algorithms may not be able to learn from them because of high sparsity, the curse of dimensionality and the 'many features; few examples' problem.

Classically, hand engineered features would be calculated from graphs for use in machine learning. For example, in link prediction, one may want to encode the strength of interactions between pairs of nodes. Alternatively, in node classification, one may want to encode information about the local and global context of nodes. Encoding this information

may be inflexible, biased, time-consuming or not suited to vector representations. Below, we introduce a variety of modern approaches to applying machine learning to graphs.

4.1.1 Guilt by association

Guilt by association relies on a notion of similarity between entities. Metadata annotated to one entity can be transferred to the other entities that are sufficiently similar, because these entities are guilty by association. This approach is employed by GO [402], where 40% of all annotations are assigned to homologous proteins [403].

Graphs can be used to predict labels using guilt by association [65, 66, 116, 117, 120], where edge weights and the network topology encode similarities between nodes. Under this framework, nodes that are either directly connected, close by in the network or appear in similar network contexts are guilty by association. Functional annotations attached to one node can be transferred to the other associated nodes. However, opponents of guilt by association approaches have shown that functional information is not encoded throughout the network, but rather is concentrated to a small number of specific edges [404].

4.1.2 Graph embeddings

Embedding methods learn representations of nodes that encode structural information about the graph [92]. Embeddings are vectors that represent a set of latent features that are automatically learnt from the graph. The goal is to optimise the embedding, such that relationships in the embedding space recapitualate relationships in the graph space. Protein network graph embeddings of have been used previously for function prediction [65, 66]. The crucial advance of graph embedding methods is that the function that maps nodes to the embedding space is learnt from the data in an unsupervised way. Graph embedding methods are not domain-specific, so they can be applied to any graph. Low-dimensional embeddings tend to be learnt, where on the order of $10^2 - 10^3$ latent dimensions are used. These embeddings are small enough such that they can be used to train off the shelf machine learning models. Alternatively embeddings can be used to calculate distances between nodes—in terms of the differences in their network contexts—with applications in clustering.

4.1.3 Encoder-decoders

Encoder-decoders are a general framework used by embedding methods [92]. In encoder-decoders, the encoder first maps nodes to low-dimensional embeddings, followed by reconstruction of the original data by the decoder, using only the embeddings. Transitively,

if the original data can be reconstructed by the encoder-decoder model, then the embeddings must contain all salient information in the graph. As such, the embeddings can be used for machine learning.

The encoder and decoder functions are learnt in an unsupervised way from the data using an optimisation process. In order to do this, a loss function must be defined to measure the difference between the original data and its reconstruction from the embeddings. The encoder and decoder functions are then optimised to minimise the reconstruction loss, and, concomitantly, the embeddings are improved.

Autoencoders are a type of encoder-decoder model implemented using an unsupervised neural network model. In this chapter, we make extensive use of deepNF, an autoencoder-based method that learns node embeddings across multiple graphs. We introduced deepNF in Section 1.5.1.2 and showed an overview of the autoencoder architecture in Fig. 1.9.

4.1.4 Contributions

Functional association data are powerful predictors of protein function. Here, we perform feature learning from protein networks using multimodal deep autoencoders (MDAEs) to embed proteins into a latent space, according to their context across multiple networks. Using these embeddings, we train supervised machine learning models to predict protein function in budding and fission yeast. We began by replicating the published performance of deepNF [66] at predicting *S. cerevisiae* protein function. Following this, we improved upon deepNF in three ways. Firstly, we showed that smaller MDAE architectures, and secondly, smaller embedding dimensions, achieve comparable performance to deepNF. Thirdly, we found that protein functions can be predicted using structured learning with the same performance as predicting each function using a separate classifier. This not only reduced training and prediction time, but also allowed non-linear correlations to be learnt between features and labels. We then applied this improved model to predict *S. pombe* protein function using structured learning. Finally, we attempted to improve the predicted protein functions by learning features from a larger set of orthogonal types of protein interactions. We take this approach forward to Chapter 5, where we predict *S. pombe* protein function in combination with phenotypic and protein evolution data.

4.2 Methods

4.2.1 Protein functional association data

Protein networks were prepared as in deepNF [66]. Interactions from the Search Tool for the Retrieval of Interacting Genes/Proteins (STRING) database [405] were used. STRING has seven interaction types:

- 'neighborhood',
- 'fusion',
- 'cooccurence',
- 'coexpression',
- 'experimental',
- 'database', and
- 'textmining' (not used in this study).

Briefly, adjacency matrices were generated by the following protocol for each of the six interaction types:

1. Protein IDs were sorted alphabetically and converted to numerical node IDs.
2. Edge weights were normalised to the interval $[0, 1]$.
3. Each adjacency matrix was scaled so that rows sum to unity.
4. Random walks with restart

$$P^{(t)} = \alpha P^{(t+1)} A + (1 - \alpha) I,$$

   was applied to each adjacency matrix A. $P^{(t)}$ is a matrix whose rows are the probabilities for a random walk from the ith protein reaching the jth protein after t steps. $\alpha = 0.98$ is the restart probability. Random walks with restart was run for $t = 3$ time steps.
5. Positive pointwise mutual information was applied to P to remove low probability edges.

STRING v9.1 [406] was used to generate *Saccharomyces cerevisiae* (*S. cerevisiae*) adjacency matrices. Despite STRING v10.5 [234] being available when this work was performed, deepNF used STRING v9.1 in [66], so we also used v9.1 so that we could compare the performance of our models to deepNF. STRING v10.5 [234] was used to generate *Schizosaccharomyces pombe* (*S. pombe*) matrices. A number of additional *S. pombe* net-

works were also prepared using the same protocol:

- Genetic interactions from BioGRID [407].
- Gene expression correlations from an *S. pombe* gene expression meta-analysis [408].
- Fission yeast phenotype ontology annotations of experimentally observed phenotypes [409]. Note that these annotations are disjoint from GO annotations [60].

4.2.2 Protein function data

For *S. cerevisiae*, Mammalian Protein-Protein Interaction Database (MIPS) terms [**Pagel2005; Ruepp2004**] were divided into three levels according to the number of proteins they are annotated to (Table 4.1). The last MIPS publication is from 2005 [61], so is very outdated. deepNF was benchmarked in [66] using MIPS annotations so that its performance could be compared to another method, Mashup [65], that also predicted MIPS annotations. As such, we chose to predict MIPS annotations for *S. cerevisiae*.

Table 4.1: ***S. cerevisiae* MIPS term annotations.**

Level	Number of proteins term is annotated to	Number of terms
1	$101 - 300$	17
2	$31 - 100$	74
3	$11 - 30$	154

As neither deepNF or Mashup benchmarked *S. pombe*, rather than use MIPS annotations, we chose to use Gene Ontology (GO) annotations because the data is more comprehensive and up-to-date. GO terms were divided into three levels per ontology (Table 4.2) using the same criteria as *S. cerevisiae* MIPS annotations. Annotations with experimental (EXP, IDA, IPI, IMP, IGI, IEP, HDA and HMP) and curated (IC and TAS) evidence codes were used.

4.2.3 Feature learning from graphs using unsupervised neural networks

We used autoencoders for unsupervised feature learning from graphs, each representing a different type of protein interaction. One method that implements this type of model is deepNF [66], which we introduced in Section 1.5.1.2. Here, we replicate deepNF's published results, improve some of its properties, and then apply it to predict *S. pombe* protein function. Briefly, deepNF uses an MDAE to learn a small number of informative features (10^2) from a large number of protein interactions ($10^4 - 10^5$). An overview of deepNF is shown in Fig. 1.9.

Table 4.2: ***S. pombe*** **GO term annotations for each of the biological process (P), molecular function (F) and cellular component (C) ontologies.**

Level	Number of proteins term is annotated to	Ontology	Number of terms
1	$101 - 300$	P	7
		F	6
		C	7
2	$31 - 100$	P	34
		F	27
		C	25
3	$11 - 30$	P	97
		F	61
		C	48

4.2.3.1 Autoencoder architecture notation

For brevity, MDAE architectures are referred to using the hidden layers in the encoding portion of the autoencoder and the number of neurons in each hidden layer. For example, the MDAE architecture used in deepNF is a '2000-600 MDAE' that embeds each protein in a 600D space using two hidden layers of 2000 neurons and 600 neurons. Let n be the number of proteins in some organism and g be the number of input network adjacency matrices. The overall architecture of the 2000-600 MDAE is

$$\text{Input: } g[n] \rightarrow g[2000] \rightarrow \text{Encoding: } 600 \rightarrow g[2000] \rightarrow \text{Output: } g[n],$$

where the decoding portion is always the mirror image of the encoding portion.

In words, the 2000-600 MDAE takes g adjacency matrices with shape $[n \times n]$ as input, processes each matrix with a separate 2000 neuron layer and passes the outputs of these g layers to a single 600 neuron encoding layer. For each of the n proteins, this layer outputs a 600D vector, which is the of each protein in a 600D space. Embeddings are then passed to g 2000 neuron layers, whose outputs are used to reconstruct the g adjacency matrices. deepNF uses $g = 6$ adjacency matrices from six types of interaction from STRING (Section 4.2.1).

4.2.3.2 Autoencoders

Autoencoders are a type of neural network that consists of an encoder and a decoder. The network tries to reconstruct the input as best it can. Reconstructions are rarely perfect, but that is not why autoencoders are used. Instead, they are used to learn a set of latent

features from the input data. To do this, constraints are imposed on the encoder part of the network that prevent a simple identity function from being learnt. In this work, we impose a size contraint on the encoder, where inputs are embedded into a low-dimensional latent space. As such, the reconstuctions will be lossy, but the embedding will be forced to be informative.

Sigmoid activations were always used on the output layer. Autoencoders were trained using data from 90% of proteins. The remaining 10% of proteins were used as a validation set to monitor training. Models were trained using binary crossentropy loss. Batch sizes of 128 examples were used. Unless specified otherwise:

- Rectified linear unit activation functions were used on hidden layers.
- The Adam optimiser [90] was used.
- Models were trained for 500 epochs, where the weights from the epoch with the lowest validation loss were used to generate the embeddings.

Models were implemented in Python v3.6 [410] using Keras v2.1.5 [411] (TensorFlow v1.8.0 [412] backend).

4.2.4 Protein function prediction

Protein functions were predicted using supervised machine learning. Models were trained to learn a mapping from protein embeddings, generated by MDAs, to protein functions.

4.2.4.1 Performance metrics

Classifier performance was measured using four metrics:

- Macro-averaged area under the precision-recall curve (MAUPR), calculated as the mean AUPR for each class separately.
- Micro-averaged area under the precision-recall curve (mAUPR), by aggregating all classes together and calculating a single AUPR.
- Accuracy, defined as $\frac{\text{TP}+\text{TN}}{|\text{P}|+|\text{N}|}$, where TP are true positives of the positive class P, and TN are true negatives of the negative class N. More specifically, the subset accuracy is calculated, where a multiclass prediction is counted as correct if and only if the predictions match the known labels across all classes.
- f1 score, defined as $\frac{2\times\text{precision}\times\text{recall}}{\text{precision}+\text{recall}}$, where precision is $\frac{\text{TP}}{\text{TP}+\text{FP}}$ and recall is $\frac{\text{TP}}{\text{TP}+\text{FN}}$.

4.2.4.2 Support vector machines

Support vector machines (SVMs; Section 1.4.7) were trained using the one-vs-rest multiclass strategy. The radial basis function kernel was used and hyperparame-

ters were estimated using an exhaustive grid search of $C = \{1, 5, 10, 50, 100\}$ and $\gamma = \{0.001, 0.005, 0.01, 0.05, 0.1\}$, selecting values that maximised mAUPR. The soft-margin penalty C is common to all SVMs and controls the penalty applied to errors, with large values producing decision boundaries with a small margin between the two classes. The radial basis function kernel parameter γ controls the influence of the data points, with high values increasing the locality of influence that the support vectors have on kernel values. Models were evaluated using 10 independent trials of 5-fold cross-validation with 80% of the data as a training set and 20% of the data as a test set. Models were implemented in Python 3.6 [410] using scikit-learn 0.19.1.

4.2.4.3 Multi-layer perceptrons

Multi-layer perceptrons (MLPs) were trained using structured learning. An MLP architecture was used that had two hidden layers of 512 and 256 neurons. Dropout was applied to the output of the first hidden layer with $P(\text{dropout}) = 0.5$. Models were trained using binary crossentropy loss and Adam optimiser. Batch sizes of 128 were used. The dropout rate and batch size were chosen empirically. Models were evaluated using 10 independent trials of 5-fold cross-validation with 80% of the data as a training set (of which 20% was used as a validation set) and 20% of the data as a test set. Loss was calculated after each epoch using the training and validation data. Overfitting was controlled by early stopping when the validation loss no longer decreased, using a patience of 20 epochs. Weights from the epoch with lowest validation loss were used. Models were implemented in Python v3.6 [410] using Keras v2.1.5 [411] (TensorFlow v1.8.0 [412] backend).

4.3 Results

Each experiment performed in this chapter had a common structure:

1. Use an MDAE for unsupervised protein feature learning from multiple protein networks.
2. Generate a small number of informative features for each protein.
3. Train a classifier to predict protein function using these features.

4.3.1 Published performance of deepNF was replicated

We replicated the results of deepNF at predicting MIPS function annotations using the published model and parameters. A 2000-600 MDAE with sigmoid activations was trained using stochastic gradient descent for 10 epochs. The reconstruction loss for the MDAE on

the training data and a validation set shows that the MDAE is not fully trained because the loss has not levelled out by the 10^{th} epoch (Fig. 4.1). Training for more epochs, until the validation loss is minimised, may improve the performance.

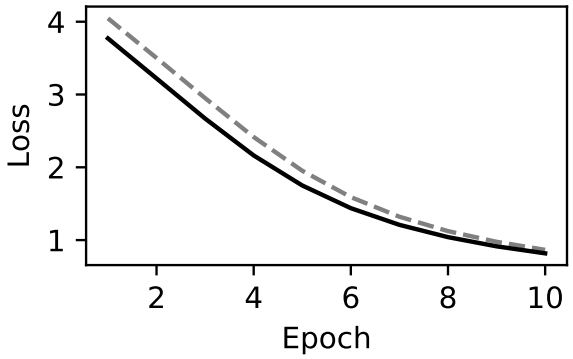


Figure 4.1: **MDAE reconstruction loss when replicating deepNF results.** Binary crossentropy loss on the validation set (solid line) and on the training set (dotted line).

The MDAE is not overfitting to the training data, so should be generalisable. When overfitting occurs, the validation loss increases, whilst the training loss continues to decrease.

When training for more epochs, overfitting should be controlled by early stopping or by saving the weights from the epoch with the lowest validation loss. Both strategies have their merits: training with early stopping is faster, but it is not possible to know whether the validation loss may begin to decrease again at a later epoch. Saving the weights from the best epoch may overcome the problem of not knowing how long to wait before early stopping, however, the model may need to be trained for a long time to be confident.

Proteins were embedded into a 600D space by the MDAE, using weights from the 10^{th} epoch. Embeddings were used as features to train SVMs to predict MIPS terms. The one-vs-rest multiclass strategy was used, where classes are treated separately and one classifier is trained per class. We successfully replicated results of deepNF published in [66] for mAUPR, MAUPR, accuracy and f1 score.

4.3.2 Small embeddings achieve comparable performance to deepNF

It is advantageous to use the smallest MDAE architecture—with the fewest number of layers, neurons and embedding dimensions—to reduce the training time of both the MDAE and classifier. However, there is no *a priori* way to choose the best neural network architecture. Instead, a variety of different architectures should be tested according to the

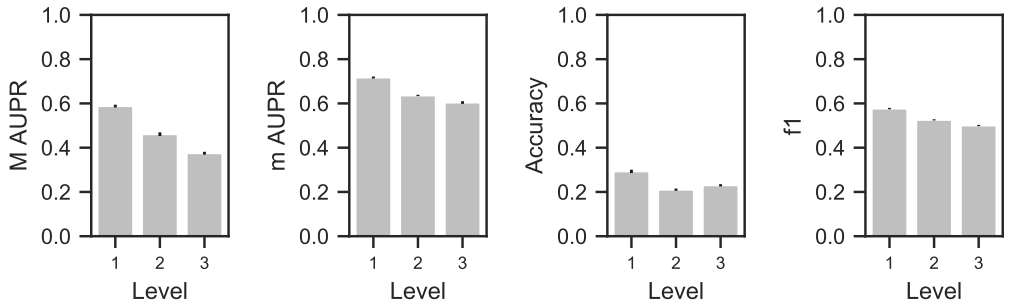


Figure 4.2: **Protein function prediction performance when replicating deepNF results for *S. cerevisiae*.** MIPS terms were divided into three levels according to the number of proteins that are annotated with each term. Bars are the mean performance across 10 independent trials of 5-fold cross-validation and error bars are the standard deviation.

following rules of thumb:

- Due to the layout of memory on the GPU, neural networks are trained best when using a power of 2 number of neurons in each layer (e.g. 128, 256, 512 and 1024).
- The output of a preceeding layer should always used as input to a layer of the same, or smaller, size (e.g. $512 \rightarrow 512 \rightarrow 128$). In autoencoders, this rule should only be applied to the encoding part of the network; decoders are the mirror image of the encoder.

To estimate the number of epochs required to train smaller MDAE architectures, we trained an arbitrary 512-128 MDAE architecture for 2000 epochs. Training for 500 epochs is sufficient because the MDAE started overfitting on the training data after ~ 150 epochs. Going forward, we train MDAs for 500 epochs and save weights from the epoch with the lowest validation loss.

We trained nine different MDAE architectures that embed proteins into a 128D or 256D space. 128D embeddings were generated by 256-128, 512-128, 1024-128, 256-256-128, 512-256-128 and 512-512-128 MDAE architectures. 256D embeddings were generated by 256-256, 512-256 and 512-512-256 MDAE architectures. The 256-256 MDAE had the lowest validation loss of 0.162 after 161 epochs (Fig. 4.3). Embeddings from this MDAE were taken forward for further experiments.

We benchmarked the predictive performance of these embeddings, using the same supervised machine learning procedure used by deepNF. We found that the 256D embeddings have an equivalent performance to the 600D embeddings used by deepNF (Fig. 4.4). This is a significant improvement over deepNF because smaller MDAs have fewer param-

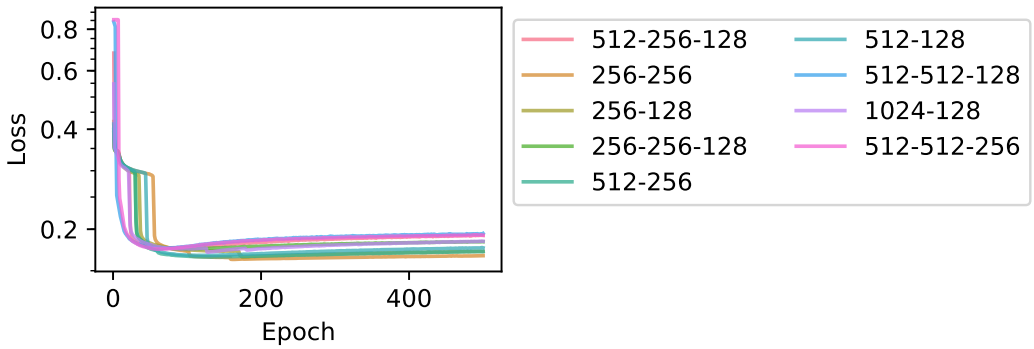


Figure 4.3: ***S. cerevisiae* MDAE reconstruction losses for different MDAE architectures that produce 128D and 256D embeddings.** Binary crossentropy loss on the validation set is plotted.

eters to train, they are less likely to be underfit on small training data sets, and they can be trained much faster. Finally, SVM training time will be faster on smaller embeddings.

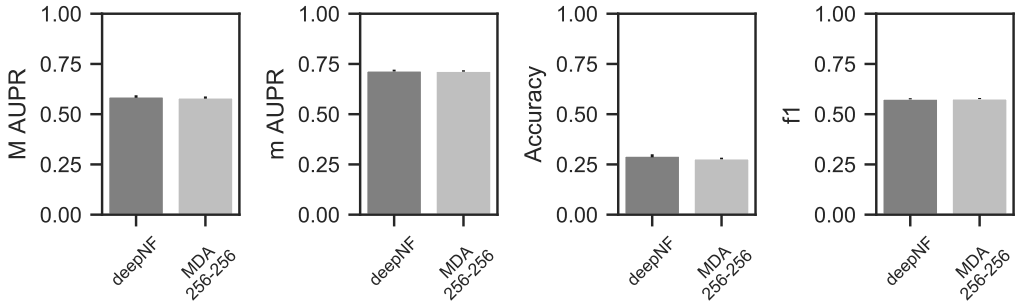


Figure 4.4: ***S. cerevisiae* protein function prediction performance on 256D embeddings.** SVMs were trained to predict MIPS terms in the one-vs-rest strategy. MIPS terms were divided into three levels according to the number of proteins that are annotated with each term. Results for level 1 terms are plotted (light grey bars). deepNF results from Fig. 4.2 are shown for comparison (dark grey bars). Bars are the mean performance across 10 independent trials of 5-fold cross-validation and error bars are the standard deviation.

4.3.3 Sets of protein functions can be predicted simultaneously using structured learning

Using our 256D embeddings, we saught to improve on the performance of deepNF. To do this, we trained neural networks instead of SVMs. Whilst most classifiers output a scalar value, neural networks output a vector. The output vector can predict all classes of a multiclass prediction task simultaneously, in a supervised learning paradigm called structured learning or structured prediction. In this paradigm, a model can be learnt to predict multiple classes that are not mutually exclusive, whilst simultaneously learning

non-linear correlations between features and labels. Output vectors can be converted to probabilities by applying the softmax function

$$S(x_i) = \frac{e^{x_i}}{\sum_j e^{y_j}}.$$

After much experimentation, we settled on an MLP with two hidden layers of 512 and 256 neurons, a dropout probability of 0.5 and a batch size of 128. The structured prediction performance of the MLP is comparable with deepNF's one-vs-rest SVM predictions (Fig. 4.5). Whilst the MLP has slightly lower performance according to mAUPR, MAUPR and accuracy, their f1 performance is higher. On balance, we believe that these moderate reductions in performance are worthwhile, due to the concomitant benefits of structured prediction and the orders of magnitude faster training time compared to one-vs-rest SVMs.

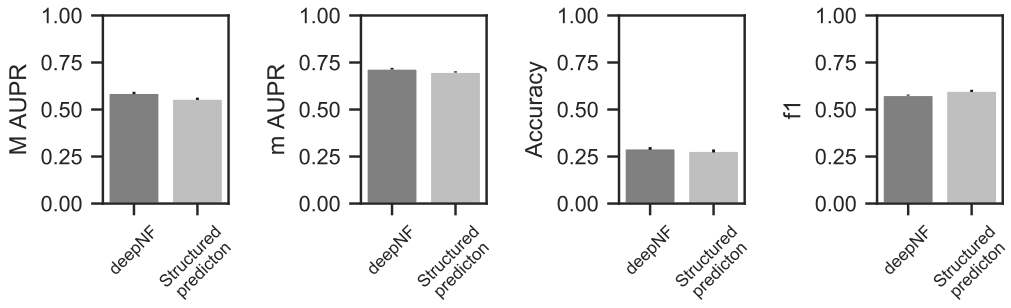


Figure 4.5: ***S. cerevisiae* protein function prediction performance using structured learning on 256D embeddings.** MLPs were trained to predict all MIPS terms in a level simultaneously. MIPS terms were divided into three levels according to the number of proteins that are annotated with each term. Results for level 1 terms are plotted (light grey bars). deepNF results from Fig. 4.2 are shown for comparison (dark grey bars). Bars are the mean performance across 10 independent trials of 5-fold cross-validation and error bars are the standard deviation.

4.3.4 Embeddings of yeast proteins predict function

With a performant protein function prediction model in hand, we next focussed our attention on predicting *S. pombe* protein functions. First, we generated embeddings for the 5100 *S. pombe* proteins contained in STRING (v10.5), using the same MDAE architectures as *S. cerevisiae*. Similar to *S. cerevisiae*, we found that the 256-256 MDAE was the best architecture, achieving a validation loss of 0.221 after 127 epochs. Validation losses were much higher for *S. pombe* than for *S. cerevisiae*, which may reflect the higher sparsity of *S. pombe* data, compared with *S. cerevisiae*, which has been studied extensively.

Using the 256D embeddings, we predicted GO term annotations from each ontology using a 512-256 MLP and structured learning (Fig. 4.6). We achieved good performance for predicting functions of proteins from level 1 for all ontologies. Level 2 terms were also predicted well according mAUPR, MAUPR and f1, but not according to accuracy. In this case, the subset accuracy is a harsh metric for multiclass prediction because the vector of predictions must exactly match the vector of labels. Biological process and molecular function terms were predicted less well than cellular component terms in level 3. Overall, the performance of predicting terms from the cellular component ontology is more consistent across the three levels.

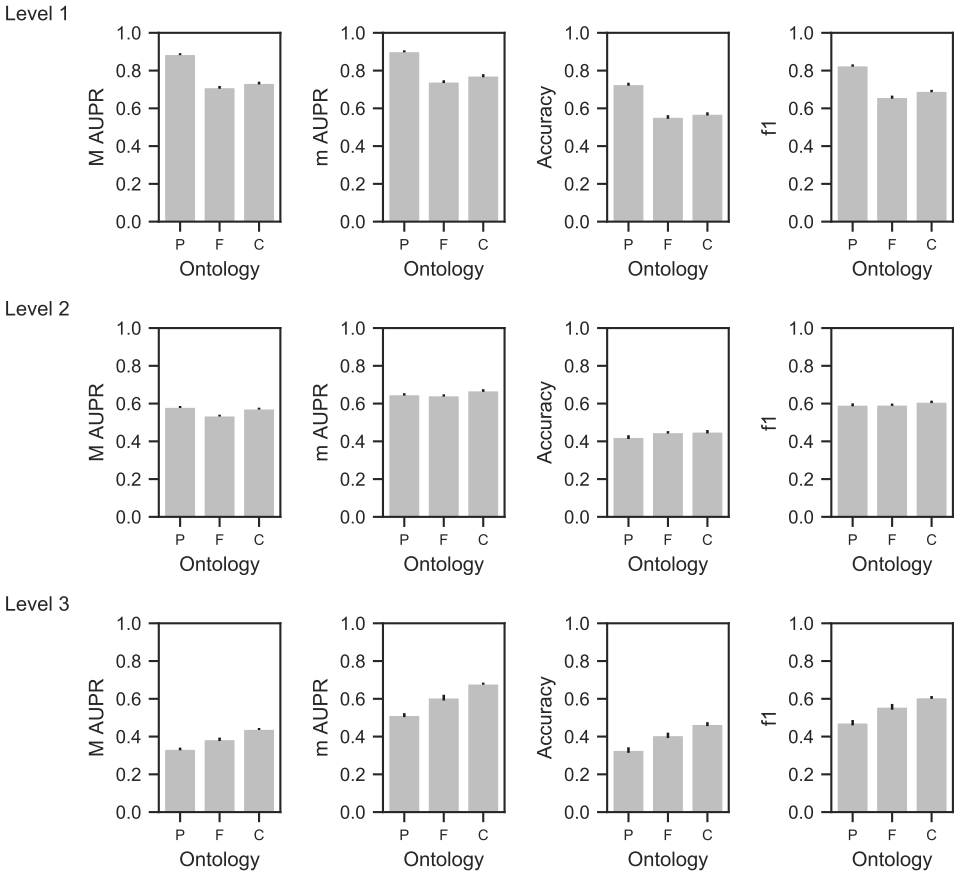


Figure 4.6: ***S. pombe* protein function prediction performance using structured learning on 256D embeddings.** MLPs were trained to predict all GO terms in a level simultaneously. GO terms were divided into three levels according to the number of proteins that are annotated with each term. Bars are the mean performance across 10 independent trials of 5-fold cross-validation and error bars are the standard deviation.

Splitting terms into three levels according to how many proteins they are annotated to seems quite arbitrary. Instead, we trained models to predict all terms from an ontology

simultaneously using structured learning (Fig. 4.7). Generally speaking, we are able to predict all terms in an ontology with approximately the same performance of predicting level 3 terms in Fig. 4.6. Whilst these trends are true for mAUPR, MAUPR and f1, it is not true for accuracy, due to a subset accuracy of ~ 0.25 for each ontology.

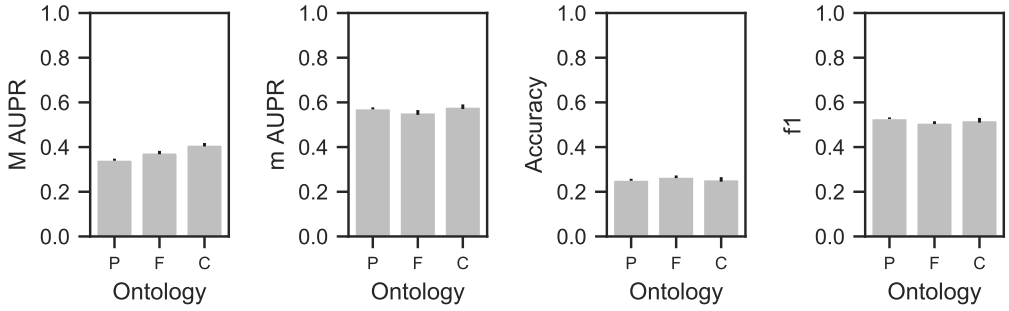


Figure 4.7: ***S. pombe* protein function prediction performance using structured learning on 256D embeddings.** MLPs were trained to predict all GO terms in an ontology simultaneously. Bars are the mean performance across 10 independent trials of 5-fold cross-validation and error bars are the standard deviation.

4.3.5 Including orthogonal protein network data did not increase prediction performance

In order to learn a low-dimensional embeddings space, MDAs perform data fusion of multple networks. The goal of data fusion is to combine multiple, heterogenous and orthogonal data sets together into a composite data set, whose predictive power is higher than any one data set alone. deepNF fused six types of protein network data encoded in the STRING database. Many more types of interactions between proteins are possible, but were not included in deepNF. Here, we tested whether the predictive power of protein embeddings could be increased by including other, orthogonal types of interactions:

- genetic interactions from the BioGRID database,
- gene co-expression correlations from a meta-analysis of *S. pombe* protein expression studies [408], and
- experimental phenotypes from the fission yeast phenotype ontology.

We fused these three networks and the six STRING networks using the same set of MDAE architectures that we used in Section 4.3.4. The 256-256 MDAE architecture had the best validation loss of 0.403. For comparison, this loss is much higher than the best loss of 0.221 when autoencoding the six STRING networks (Section 4.3.4). This suggests that the three additional networks are noisy, reducing the MDAE's ability to reconstruct

the networks. An alternative argument to explain this result could be that the reconstruction loss is higher for nine input networks because the 256-256 MDAE's neurons were saturated and the MDAE has reached its maximum reconstruction capacity. However, this situation is unlikely to be true because 31 embedding dimensions remain untrained in the MDAE, where all values in these dimensions are zero for all proteins. Also, other architectures that were larger, such as the 512-512-256 MDAE, had very similar loss curves. Either way, the 256D embeddings generated by feature learning from nine networks had a lower prediction performance (Fig. 4.8) than 256D embeddings from six STRING networks (Fig. 4.6).

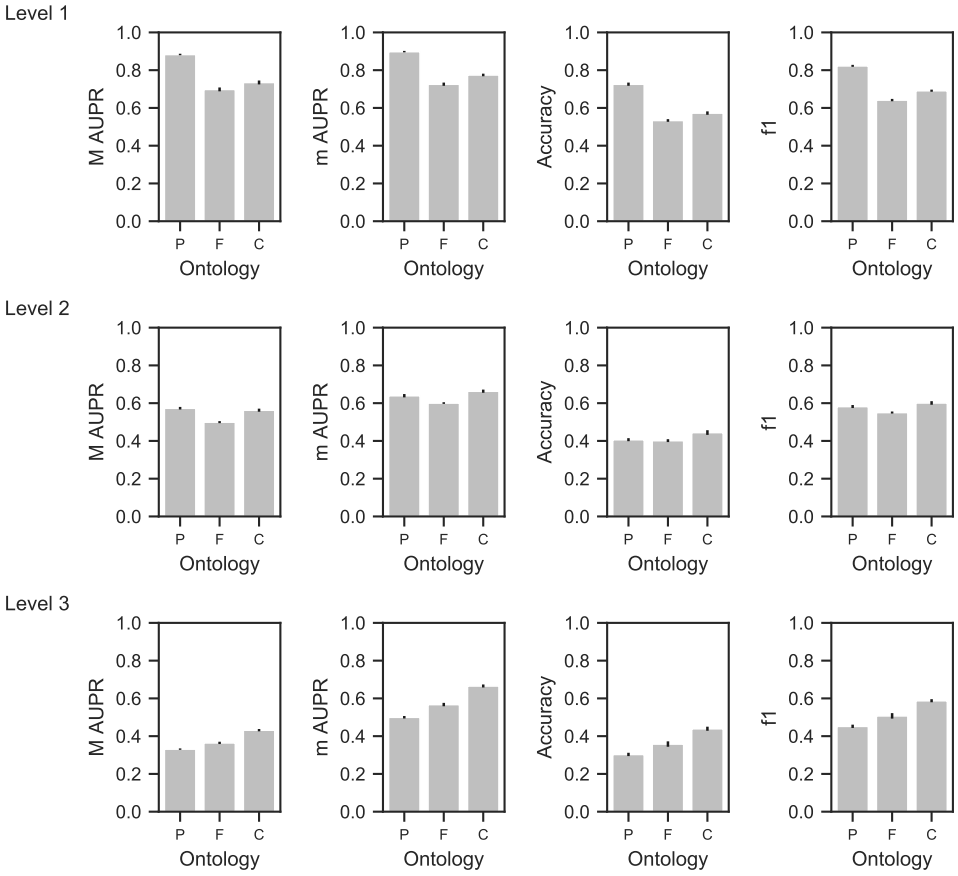


Figure 4.8: ***S. pombe* protein function prediction performance on 256D embeddings generated by feature learning from nine networks.** MLPs were trained to predict all GO terms in a level simultaneously. GO terms were divided into three levels according to the number of proteins that are annotated with each term. Bars are the mean performance across 10 independent trials of 5-fold cross-validation and error bars are the standard deviation.

We also tried fusing three other combinations of data, but could not improve upon the performance of six STRING networks alone. However, we chose to not show these

results because they are similar to Fig. 4.8 and are highly repetitive We tested the following combinations:

1. six STRING networks, genetic interaction and gene co-expression,
2. five STRING (not co-expression), genetic interaction and gene co-expression, and
3. STRING physical interaction network, genetic interaction and gene co-expression.

Combinations 1 and 2 produced slightly lower prediction performance than the six STRING networks. Combination 3 yielded much worse performance than using six STRING networks.

4.4 Discussion

4.4.1 Small embeddings achieve comparable performance to deepNF

deepNF used a 2000-600 MDAE, which contains an unnecessarily large number of parameters to model the amount of information contained in *S. cerevisiae* protein networks. Models of this size require $\sim 10^6$ examples to train all parameters sufficiently. Because we only have in the order of $10^3 - 10^4$ proteins as examples, smaller architectures should be used. We tested whether smaller embeddings of 128D or 256D could be used (Section 4.3.2). We found that 256D were sufficient to replicate the performance of deepNF's 600D embeddings (Fig. 4.4). Although we were unable to surpass the published performance of deepNF, this is still an improvement over deepNF. By using smaller embeddings, MDAs, SVMs and MLPs can be trained faster, which is beneficial when performing research using machine learning.

4.4.2 Sets of protein functions can be predicted simultaneously using structured learning

Neural networks can be trained using structured learning, where models output vectors, rather than scalars. In structured learning, models can predict multiple classes simultaneously, whilst also learning non-linear correlations between sets of features and sets of labels.

We used structured learning to predict *S. cerevisiae* protein function (Section 4.3.3). We found that structured learning predicts function using 256D embeddings with comparable performance to deepNF using 600D embeddings and one-vs-rest SVMs (Fig. 4.5). On balance, we believe that these moderate reductions in performance are worthwhile, due to the concomitant benefits of structured prediction and the orders of magnitude faster

training time compared to one-vs-rest SVMs.

4.4.3 Embeddings of yeast proteins predict function

We applied this model to predict *S. pombe* protein function using 256D embeddings and structured learning (Section 4.3.4). We either predicted all terms from an ontology simultaneously, or split terms from each ontology into three levels and predicted separately all terms from each level simultaneously.

For structured learning of levels within an ontology, we predicted cellular component terms consistently well in all three levels. This is probably due to the wider coverage of experimental annotations for these terms. Cellular component terms can be experimentally validated easily using simple assays and high-throughput methods. Conversely, the predictive performance for the biological process ontology is high for level 1, but drops steeply by level 3, which may reflect the comparatively smaller number of proteins with experimental annotations. These conclusions are recapitulated in an analysis that we conducted into the distribution and quality of *S. pombe* GO annotations in Section 5.3.1. For structured learning of entire ontologies, we obtained a performance approximately the same as the performance for level 3 terms from each ontology, respectively.

Subset accuracy is clearly an inappropriate metric for structured learning tasks. It is much too strict when predicting a vector of labels. The other three metrics, however, hold up well under structured learning.

Going forward, one strategy for protein function prediction could be to train an ensemble model, consisting of a structured learning model and a one-vs-rest model. Classifiers trained using the one-vs-rest strategy would learn patterns that predict individual functions well. On the other hand, structured learning models would learn non-linear correlations between features and labels to predict subtle, complex functions. The ensemble model E could combine predictions from the one-vs-rest model O and the structured learning model S using a strategy such as:

- Logical OR: $\vee(O = 0, S = 1) = 1$
- Max: $\max(O = 0.9, S = 0.4) = 0.9$
- Mean: $\text{mean}(O = 0.9, S = 0.4) = 0.65$

4.4.4 Including orthogonal protein network data did not increase prediction performance

We tested whether the performance of predicting *S. pombe* protein functions could be improved by including additional, orthogonal protein network data. We trained MDAs to learn features from different combinations of network data, with and without the six STRING networks used by deepNF. In all cases, we were unable to improve upon the features learnt from six STRING networks. Compared to the six STRING networks alone, the reconstruction losses for MDAs trained on different combinations of networks were much higher. Furthermore, the protein function prediction performance was also much lower when using these different combinations of networks, compared to the six STRING networks alone.

It is unclear why this is the case. STRING is a trusted resource that is curated to a high-quality. The six STRING networks may have been processed in some way, such that the six networks correlate well with each other. Including exogenous data, such as genetic interactions, gene co-expression correlations and experimental phenotypes may not correlate well with the STRING data. Alternatively, these other three types of interaction data may be more noisy than STRING data, due to the nature of the data, how it was processed, or the level to which the database has been curated.

In the future, it will be interesting to see whether additional, orthogonal types of interaction data can be fused together successfully. Protein functional association resources continue to improve as measurement accuracies increase, large-scale experiments become commonplace and previous experimental results are independently validated.

4.4.5 Conclusion

The work in this chapter successfully replicated the published performance of a novel protein function prediction method, deepNF, that uses MDAs and protein network data to generate embeddings of proteins. We then improved upon deepNF in three ways. Firstly, we showed that smaller MDAE architectures, and secondly, smaller embedding dimensions, achieve comparable performance to deepNF. Thirdly, we found that protein functions can be predicted using structured learning with the same performance as predicting each function using a separate classifier. This reduced training and prediction time, whilst also allowing non-linear correlations to be learnt between features and labels. We applied our improved model to successfully predict *S. pombe* protein function using structured

learning. We use this approach in Chapter 5, in combination with phenotypic and protein evolution data, to predict *S. pombe* protein function.

Chapter 5

Learning the functions of fission yeast proteins

5.1 Introduction

In this chapter, we develop a machine learning model to predict protein function using a combination of network, evolutionary and phenomics data. We train machine learning models using protein network embeddings from Chapter 4, evolutionary family data from CATH-FunFams and phenomic data from high-throughput phenomics screens of *Schizosaccharomyces pombe* (*S. pombe*; fission yeast) gene deletion mutants. First we analyse the phenomics data, then rigorously benchmark the machine learning models, before predicting functions of *S. pombe* proteins. Finally, we enter our predictions into the Critical Assessment of Functional Annotation (CAFA) evaluation of protein function prediction methods. We begin the Introduction with pertinent information about fission yeast, phenomics screening and CAFA.

5.1.1 *Schizosaccharomyces pombe*

S. pombe is a unicellular eukaryotic species from the Fungi kingdom, and is *probably the best model organism in the world* [413]. Fission yeast was first described in 'pombe', the Swahili word for a type of beer produced in East Africa. The standard laboratory strain was isolated from wine grapes in French viticulture [414]. The natural habitat of fission yeast is not known, but it is now found throughout the world in association with many human activities [414].

The 'other' yeast *Saccharomyces cerevisiae* (*S. cerevisiae*; budding yeast) replicates asexually by budding—an asymmetric variant of mitosis [415]. Whilst *S. pombe* also replicates asexually, its cells divide symmetrically by binary fission in a process that is more

similar to mammalian mitosis [416]. *S. pombe* is usually haploid, but, in stressful situations, two haploid cells of opposite mating types can fuse to form a diploid cell, which then divides into four haploid spores [417].

The *S. pombe* genome is composed of 13.8 Mb of DNA across three chromosomes [418]. Phylogenetics reveals that fission yeast diverged from budding yeast 330 to 420 million years ago, which makes the two yeasts as different to each other as animals are to either [419]. Fission yeast has 338 genes that are conserved in humans, but absent in budding yeast, which is almost twice as many as the 179 budding yeast genes that have orthologs in humans, but are absent in fission yeast [416]. As such, fission yeast is an ideal model of eukaryotic cellular processes [420].

Functional genomics in *S. pombe* has been aided by the generation of a library of haploid gene deletion strains for all non-essential genes [421]. Each gene was replaced with a selectable KanMX marker that allows for positive selection of the knockout strains. Short oligonucleotide sequences, known as barcodes, flank the marker with a unique sequence for each knocked out gene. All barcodes can be amplified by the polymerase chain reaction simultaneously using universal sequences. This allows strains to be profiled in a parallel and competitive fashion using Bar-seq [422, 423], or barcode sequencing.

5.1.1.1 Function annotations

Since 1992, when the budding yeast genome was sequenced, the percentage of functional characterisation of its proteome initially increased rapidly, but over the past decade has plateaued at 82% [424]. Fission yeast saw a much faster increase over a shorter time frame, but it too has topped out at 84% coverage [424]. Biological roles for the remaining $\sim$ 20% of proteins remain elusive. Strangely, many of these proteins are conserved in humans, which suggests that they are involved in key cellular processes and therefore are a priority to study [424]. There are many reasons why a protein may not have been studied [425], ranging from biological biases (experimental assays, detectability of functions, cost effectiveness) to cultural biases (funding priorities, citability, fashion trends). Next, we touch on some explanations for these biases.

Yeast gene deletion mutants that are unable to grow in standard laboratory conditions on rich media are deemed to be 'essential' genes. In fission yeast, there are 1,390 essential genes, found by searching PomBase [426] for 'inviable cell population' (Fission Yeast Phenotype Onotology (FYPO) [409] term FYPO:0002059; null expression, single allele

genotypes) on March 23, 2020. Of the non-essential genes in budding yeast, 34% of gene deletion mutant strains show a growth phenotype when grown in standard conditions, but 97% of genes are essential for normal growth in at least one of $1{,}144$ different chemical genomic assays [427]. The remaining 3% of genes that did not display a phenotype may do so in at least one condition that has not (yet) been tested. Experimental function annotations are known to be biased towards high-throughput assays that are able to generate large volumes of functional annotations for many proteins (see Section 5.1.2 for an example), but usually only for a tight range of functions [428]. Furthermore, these annotations are usually subject to the curse of 'few articles, many proteins', whereby 0.14% of papers in the Gene Ontology Consortium account for the annotations of 25% of proteins [428]. Functional annotations derived from high-throughput experiments have a high error rate, so would benefit from being confirmed by independent high-throughput studies, or targeted low-throughput experimental validation. As such, the degree to which we can trust functional annotations, across all species, is questionable. We explore the scope of opportunities for protein function prediction in *S. pombe* in Section 5.3.1 by analysing Gene Ontology terms annotated to proteins. Finally, the law of diminishing returns suggests that publishing papers on highly studied proteins such as p53 (that has amassed over $33{,}000$ publications since 2007 [424]) will not be as fruitful as investigating a conserved human protein whose orthologues in any species have not been characterised.

5.1.1.2 Function prediction

A previous joint PhD project between the Orengo and Bähler groups involved the development of Compass, a method to predict *S. pombe* protein function using protein networks. Using the *S. pombe* Search Tool for the Retrieval of Interacting Genes/Proteins (STRING) network [234], weighted adjacency matrices were generated for each of the seven edge types: 'neighborhood', 'fusion', 'cooccurence', 'coexpression', 'experimental', 'database' and 'textmining' (see Section 4.2.1 for details). These adjacency matrices were summed to form a combined graph—a cheap and cheerful data fusion technique. To account for false negative and false positive edges, a kernel function was applied to the adjacency matrix to generate a kernel matrix. In this case, the commute-time kernel [115] was generated, which represents the expected number of steps it would take to travel from node v_i to v_j and return back to v_i in a random walk on graph nodes. The commute-time is not only able to measure distances between nodes, but is also able to capture topological features of

the graph's wiring pattern at mesoscopic resolution. This is beneficial because the Bähler group has previously shown that topological features of *S. pombe* functional networks are predictive of protein function [429, 430]. Network commute-times, and kernels thereof, were used previously by the Orengo group to successfully predict protein function in a guilt-by-association framework using combinations of kernels to fuse heterogenous data and kernel matrices to measure the similarity between pairs of proteins [116, 431]. Lehtinen *et al.* [117] used kernel matrices to train a supervised partial least squares regression model [102] to predict protein function by supervised learning. Partial least squares regression first learns a low-dimensional representation of the kernel matrix, in a similar manner to dimensionality reduction into principal components, followed by ordinary least squares regression in this low-dimensional space.

Compass was benchmarked against GeneMANIA [108], the de facto method for protein function prediction at that time. GeneMANIA is a network-based protein function prediction method. First, GeneMANIA combines multiple functional association networks by calculating a weighted average of the adjacency matrices. A ridge regression model (L2 regularised linear regression) is trained to learn the weightings for each network, with the ability to downweight redundant and irrelevant information. Then, GeneMANIA runs label propagation on the composite network to predict protein function. The label propagation algorithm takes a set of seed proteins, a set of positive examples (positive node bias), and optionally a set of negative examples (negative node bias). Labels are propagated through the network in an optimisation process guided by a cost function that tries to minimise the difference between the label of neighbouring nodes and also the difference between the predicted label and the initial bias for each node. Compass consistently outperformed GeneMANIA on a variety of benchmarks and across a panel of metrics [117].

The field of protein function prediction has seen rapid development and progress in the past decade, enabled by higher-quality data that covers more proteins and functions, and on the other hand by advances in machine learning methods and computational power, which together have produced more performant models. We make use of these advances in this study to make better predictions of *S. pombe* protein function.

5.1.2 Phenomics

Phenomics is the description and prediction of phenotypes in relation to genotypes [432]. Use of the term took off in 1999, when Schilling and colleagues explained that genome se-

quencing, proteomics, systems biology and bioinformatics have enabled large, heterogenous data sets to be interrogated to understand genotype-phenotype relationships [433, 434]. That year, an early commentary about the nascent field of proteomics suggested that phenomics is "an all embracing term (on a par with genomics) to describe functional genomics" [435]. Phenomics screens are typically genome-wide high-throughput experiments that produce high-dimensional phenotypic data.

Some phenotypes have direct one-to-one relationships with their causal genotypes. Examples include fully penetrant mutations in the amyloid-β precursor protein that cause familial Alzheimer's disease (Section 2.1.1), or Gregor Mendel's genetic hybridisation experiments in peas [436]. However, the majority of phenotypes are complex (meaning 'multivariate', where more than one gene determines a phenotype) where there is no clear relationship between genotype and phenotype. It cannot be stated more succinctly than in [437]

> Although genotypes exist and are inherited in a discrete space convenient for many sorts of analyses, the causation of key phenomena such as natural selection and disease takes place in a continuous phenotype space whose relationship to the genotype space is only dimly grasped.

Obvious exemplary complex phenotypes include height, susceptibility to cancer, and rate of ageing. Despite the lack of a clear relationship between genotype and phenotype, it is clear that there *is* a relationship.

Another way to think about complex phenotypes is that they are emergent properties [438]. Genotypes gives rise to phenotypic units that are integrated, from which a complex phenotype may arise [439]. Novikoff states in [438] that

> Knowledge of the laws of the lower level is necessary for a full understanding of the higher level; yet the unique properties of phenomena at the higher level can not be predicted, a priori, from the laws of the lower level.

In other words, phenomics can elicit higher level phenotypes and can predict which lower levels (genes or variants) should be investigated further.

Phenomics has become a popular tool in the geneticist's toolbox. For example, it has been used to study the effects of obesity in human [440], increase crop yields [441, 442], and understand mammalian gene function in mouse [443]. Over the next decade, many

more genotype-phenotype relationships will need to be untangled now that gene editing has been trivialised with CRISPR-Cas9 [444, 445]. Phenomics will likely play a key role [446].

Phenomics has also been an important tool for studying genotype-phenotype relationships in *S. pombe* [447]. Genome-wide screens in fission yeast are facilitated by the Bioneer library of gene deletion strains for all non-essential genes [421] and the ease with which quantitative phenotypes can be collected. A number of phenotypic proxies are used in *S. pombe*, including automated colony-sizing of colonies grown on solid media with computational processing of the large volume of data [448–451], or the abundance of barcodes in multiplexed Bar-seq experiments [423, 452].

A common strategy is to screen strains in the presence of some stressor that elicits the conditional essentiality of genes. One study identified 33 new ageing genes by inhibiting the TORC1 signalling pathway in the gene deletion mutant stain library with rapamycin and caffeine [453]. Whilst it was known that a high dosage of caffeine is toxic to fission yeast, but a low dosage can be tolerated [454, 455], the genetic reasons were unknown. A second screen of the gene deletion mutant stain library in media containing caffeine found that oxidative stress pathways are responsible for tolerance [456]. A third study identified 60 new genes by screening the gene deletion mutant strain library in the presence of the transcriptional inhibitor 6-azauracil [457].

5.1.3 CAFA 4

The latest iteration of the CAFA protein function prediction challenge [72–74], CAFA 4, took place between October 2019 and February 2020. CAFA's organisers provide a set of target protein sequences, where the aim is for participants to predict the functions of these sequences. In CAFA4, the targets were whole proteomes from 18 model organism species, including human, mouse, fly, worm, baker's yeast, and, crucially, fission yeast. Participants could predict functions from any of three disjoint ontologies: Gene Ontology, Human Phenotype Ontology [63], and Disorder Ontology [64], which was included for the first time. Viewed in its wider context, CAFA 4 ultimately aimed to improve the quality and coverage of functional annotations for a large number of important proteins in the biological sciences. Each participating team could enter predictions from three separate models, which were assessed for coverage using $F_{\max}$ (Eq. (1.1)) and (semantic) precision using $S_{\min}$ (Eq. (1.2)). Teams were ranked according to their best performing model on

each metric, which allows teams to enter a 'strict' model that is optimised for $S_{\min}$ and a 'relaxed' model for $F_{\max}$.

An initial evaluation of CAFA 4 took place and was presented at the ISMB conference in July 2020. The performance of the top 10 models under $F_{\max}$ and $S_{\min}$ were presented for each of the three Gene Ontology (GO) name spaces. The final evaluation is expected to take place in November 2020, with a publication expected in mid-2021.

5.1.3.1 Contributions

Fission yeast is an important model organism to understand the biological mechanisms of cellular processes. In this chapter, develop machine learning models that predict the functions of *S. pombe* proteins. To do this, we collected a large, phenomic data set of *S. pombe* gene deletion mutant strains grown in 131 different conditions. Using an extensive array of analyses, we consistently found that the growth phenotypes were not reliable, reproducible or predictive of protein function. We trained machine learning models using this bespoke experimental data, alongside orthogonal data from protein networks and evolutionary information from CATH. We evaluated the performance of models trained on these data modes separately, and in combination, finding that the best performing model used a combination of network and evolutionary data. In particular, we focussed on functions that were predicted for proteins conserved in vertebrates, and those that currently have no experimentally characterised functions in *S. pombe*. Finally, we entered the predictions from this model into the fourth CAFA evaluation of protein function prediction methods.

5.2 Methods

5.2.1 Growth phenotyping

5.2.1.1 Collection

All genes have been systematically deleted from *S. pombe*. Strains of the $\sim$ 3,700 non-essential gene deletion mutants (out of 5,137 genes in total) are available commercially as a library, known as the Bioneer library [421].

To measure the effect of gene deletions on the growth phenotype of *S. pombe*, the library was plated onto agar media supplemented with different molecules. The plates were incubated at various temperatures and for various times to allow the colonies to grow. The particular combination of media, added molecules, temperature and incubation time is referred to as the 'condition'. Growth phenotypes were collected from 131 unique

conditions. At an appropriate time point—when colony growth is not saturated, so that slow-growing colonies can be distinguished from fast-growing colonies—the plates are scanned.

The library was grown in $1{,}536$ colony format, spread across three plates. Plates exhibit spatial biases, especially at the edges where competition for resources is lower. Wild type strains are plated every four strains to form a grid that is used to correct colony sizes and remove spatial biases.

5.2.1.2 Processing

Plate images were processed using pyphe [450]. pyphe was inspired by scan-o-matic [449], a pipeline for processing plate-based high-throughput growth phenotyping experiments.

The grid of wild type colonies is used to interpolate a 'reference surface' across the plate, corresponding to the expected growth of a wild type colony at any point on the plate. Colony sizes are normalised by dividing the colony size by the reference surface value for the colony's position on the plate. Colonies of gene deletion mutants that are less fit than wild type will have a relative colony size < 1, whilst those that are fitter will have a relative colony size > 1.

In total, $2{,}832{,}384$ colonies were present in the data set (Fig. 5.2). Colonies were removed from the data set if:

- gene ID is 'grid' or 'empty'
- colony size, reference surface, or colony circularity is missing
- colony size, corrected colony size, or reference surface ≤ 0

$2{,}389{,}858$ colonies remained after 'grid' and 'empty' colonies were removed (Fig. 5.2). $2{,}256{,}475$ colonies remain after poor-quality colonies were removed. Sizes were rounded to 3 decimal places to remove excess numerical precision that is probably experimental noise. Phloxine B is a red food dye that is used in yeast cell-viability assays [458]. Cell membranes prevent the dye from entering cells, however, when cells die and membrane integrity is lost, the dye enters cells. Colonies that were grown in conditions with, and without, phloxine B were pooled because phloxine B does not interfere with growth. Colony sizes for each strain were normalised, to remove effects that knocking out the gene may have on growth, by dividing by the mean colony size of the strain in plain media at $32°$C: 'YES_32' for conditions based on YES media, or 'EMM_32' for EMM media. Colony sizes were $\log_2$-transformed to make inverse growth differences symmetric about 0. For exam-

ple, $\log_2(2) = 1$ and $\log_2(2^{-1}) = \log_2(0.5) = -1$. Each strain-condition pair was then represented by the colony whose size that was most different to the wild type by

$$X[\text{argmax}(\text{abs}(X))]. \tag{5.1}$$

Finally, for each condition, any missing colony sizes were imputed using the mean colony size of strains in that condition. Mean imputation was chosen so that imputed values would not have any effect when training machine learning models. From now on, we use the term 'colony size' to refer to colony sizes that are normalised and $\log_2$-transformed for brevity.

5.2.1.3 Significance testing

Phenotype hits were called using null hypothesis significance testing. Absence of knocked out genes may affect growth—often negatively—regardless of the condition. To account for these effects, the null distribution for each strain is the distribution of its colony sizes in plain media at 32°C: 'YES_32' for conditions based on YES media, or 'EMM_32' for EMM media. For each strain and condition, the colony size distribution was compared to its corresponding null distribution using the two-sample unequal variance t-test (also known as Welch's test). The null hypothesis was that there is no difference between how a strain grows in a condition and its corresponding control condition.

P values were corrected for multiple testing using the Benjamini-Hochberg method [459], which controls the false discovery rate (FDR), i.e. the expected proportion of rejected H_0s that are true. FDR methods provide less strict control of type I errors—rejection of a true H_0, a false positive—but have higher power—the probability that H_0 is rejected when H_1 is true. The Benjamini-Hochberg method first sorts the P values $P_1, P_2, ..., P_J$. For a given significance threshold α, the first j hypotheses are rejected for which $P_j \leq \frac{\alpha j}{J}$.

The Bonferroni method [460, 461] controls the family-wise error rate (FWER), i.e. the probability of making at least one type I error (false positive rate). The Bonferroni method rejects all hypotheses for which $P_j < \frac{\alpha}{J}$. For J hypothesis tests, the FWER at a given significance threshold α is $1 - (1 - \alpha)^J$. For example, at the 0.01 significance level, if 100 tests are performed, the FWER $= 1 - 0.99^{100} = 0.634$. In other words, there is a 63% chance of making one or more type I errors. FWER methods provide strict control of type I errors, but have correspondingly low power. If it is desirable for the false positive rate (the type I error rate) to be low—for example hits are to be analysed manually—Bonferroni

could be used. Conversely, if high power is more important, Benjamini-Hochberg could be used.

5.2.2 Machine learning

5.2.2.1 GO Slim annotations

The Gene Ontology [462] '2018-11-12' release was downloaded on November 14, 2018. *S. pombe* GO annotation data '10/15/2018' release was downloaded from PomBase [426] on November 14, 2018. The 53 *S. pombe* GO Slim terms were accessed from PomBase on November 14, 2018 (https://www.pombase.org/browse-curation/fission-yeast-go-slim-terms). GO Slim terms were chosen as targets, as these terms are chosen to be sufficiently informative terms in the ontology.

The GO annotation targets for machine learning were constructed as an $[n$ genes $\times$ m terms$]$ binary matrix. For each gene and GO Slim term pair, the corresponding position in the matrix is set to 1 if the gene is annotated with the term, or any of its descendant (more specific) terms in the ontology, else 0. The following relationships were considered: 'is_a', 'part_of', 'regulates', 'positively_regulates' and 'negatively_regulates'. Only high-quality annotations with experimental (EXP, IDA, IPI, IMP, IGI or IEP) or curated (ISS, ISO, ISA, ISM, IGC, IBA, IBD, IKR, IRD, RCA, TAS or IC) evidence codes were considered. To avoid circularity, automatically assigned annotations with evidence code IEA were not considered.

For each of the three ontologies—biological process, molecular function and cellular component—the numbers of genes that have at least one annotation were counted. We consider there to be a hierarchy of preference for evidence codes of GO term annotations, starting with the 'experimental' class (consisting of the evidence codes EXP, IDA, IPI, IMP, IGI, IEP, HTP, HDA, HMP, HGI and HEP), followed by 'curated' (ISS, ISO, ISA, ISM, IGC, IBA, IBD, IKR, IRD, RCA, TAS and IC), 'automatic' (IEA), and ending with undesirable annotations from the 'bad' class (NAS and ND). We assign genes to evidence code classes using a fast and frugal tree [463, 464], that asks a total of n questions and has $n + 1$ exits—as opposed to classical decision trees that have n^2 exits [464]. For each gene, g, we begin at the first internal node of the tree by asking the question "does g have at least one 'experimental' annotation?" If so, we assign it to the 'experimental' class and exit the tree; if not, we proceed sequentially to the 'curated', 'automatic' and 'bad' classes, exiting the tree if the question at each internal node evaluates to true.

5.2.2.2 FunFam assignments

All 5,137 protein sequences encoded by the *S. pombe* genome were downloaded from PomBase [426] in FASTA format on February 21, 2019. Proteins were assigned to FunFams [54] by searching their sequences against the FunFam hidden Markov model (HMM) library using hmmsearch from HMMER3 [27] and a threshold of E $< 10^{-3}$. In total, 3,319 sequences had 149,099 hits to 23,900 FunFams.

Expect values (E-values) were $-\log_{10}$-transformed to convert to a linear scale. Multiplying by -1 means that good FunFam hits with low E-values have high feature values and weak hits with high E-values have low feature values—although the sign of features is not important for the random forest classifier used in this work. For machine learning, these data were converted to an $[n$ proteins $\times\ m$ FunFams$]$ matrix, with 99.8% sparsity, $(1 - \frac{149{,}099}{3{,}319 * 23{,}900}) * 100$.

Because 23,901 FunFams were hit, the feature matrix is very wide—at least from a machine learning perspective. So, when predicting some GO term g, models were trained on FunFams that contain at least one protein annotated with g. GO [462] annotations for all UniProt [360] accessions contained in FunFams were downloaded in February 2020. All annotations were included, except those with NAS, ND, TAS or IEA evidence codes, but UniProtKB-kw IEA curated terms were included. GO terms were associated with FunFams by identifying all GO terms that are annotated to proteins in each FunFam. Ancestor terms that have 'is_a', 'has_part', 'part_of' and 'regulates' relationships were also included.

When predicting GO terms using only FunFam data, the 23,900 FunFams that had at least one *S. pombe* homologue were used as features.

5.2.2.3 Network embeddings

Network embeddings of *S. pombe* genes in networks from the STRING [405] database were produced by a multimodal deep autoencoder (see Chapter 4 and Section 4.3.4 for details). The 256D embeddings were used from the 256-256 architecture that produced the lowest validation loss. Machine learning models were trained using the values in each dimension of the embedding as features.

5.2.2.4 Random forests

Supervised machine learning was used to predict GO annotations for *S. pombe* proteins using random forests (RF) [465]. For an introduction to RFs, see (Section 1.4.6). The RF implementation used in DecisionTree.jl builds forests of classification and regression trees

(CART). We initially experimented with 100 trees and later increased this to 500 trees for final model evaluation. There is usually only a mild improvement in the performance when increasing from 100 to 500 trees (< 0.1 AUPR improvement). The cost function used in this study employed the following criteria:

- No maximum depth, so trees could grow arbitrarily deep.
- A minimum of two samples is needed to split a node, resulting in terminal nodes with single samples in each.
- No minimum purity increase, here defined according to minimising entropy.

Trees were not pruned in this study.

To better compare the performance using different input data, we implemented a strategy to obtain the same train-tests splits for each label and repeat, regardless of the input data. Before predicting the ith label in the nth repeat, the pseudo-random number generator is seeded with $i + n$ before being used to generate the cross-validation splits. However, if the input data does not contain the same set of strains (rows) as other input data, then the splitting strategy will not be the same.

Hyperparameters were chosen using an exhaustive grid search over the parameter ranges. Each combination of parameters was assessed using a nested five-fold stratified cross-validation. The parameter combination that produced the highest area under the precision-recall curve (AUPR) was chosen is shown in Table 5.1.

Table 5.1: **RF hyperparameter optimisation.** Hyperparameters that were optimised and the parameter ranges over which an exhaustive grid search was carried out are listed.

Hyperparameter	Description	Parameter range
n_subfeatures	Number of features used to grow each tree	$\{10, 25, 50, \sqrt{n}\}$
partial_sampling	Proportion of examples used to grow each tree	$\{0.50, 0.75, 1.00\}$

Model performance was estimated using five-fold stratified cross-validation. The data was shuffled before each cross-validation. Five independent runs of cross-validation were performed to estimate the model performance under different train-test splits. Terms were predicted using the one-vs-rest multiclass strategy.

5.2.2.5 Prediction

After benchmarking, GO term annotations were predicted for *S. pombe* genes. We took two approaches to do this. Initially, we used the canonical approach of training a model using all available training data, then predicting labels. Latterly, after realising that the

random forest models had overfit on the training data and were conservative at predicting positive labels, we implemented a more liberal prediction strategy based on 5-fold cross-validation. Under this strategy, models were trained using 80% of the data and used to predict labels for the remaining 20%, and so on for each of the five train-test splits. This means that models are never trained on the same examples that they predict labels for, which forces models to predict labels using general principles that were learned in the training phase.

GO annotations were propagated to their parent terms in the GO DAG. Terms annotated to between 50 and $1{,}000$ proteins were included in the target set, so that sufficient training examples were present for machine learning. Random forest classifiers were trained using 500 trees. Because we did not use growth phenotype features to make final predictions, we did not exclude any genes from our data set. We observed that models were robust to values of hyperparameters, so for the sake of expediency, we did not perform hyperparameter optimisation. We chose sensible default values for the number of sub-features as $\sqrt{n}$ and partial sampling of 0.7. GO terms were predicted using 5-fold cross-validation. Predictive performance was evaluated using AUPR.

5.2.2.6 CAFA

CAFA is a protein function prediction challenge. Though similar to how GO terms were predicted for fission yeast, as explained in Section 5.2.2.5, a number of updates and modifications were made. STRING v11.0 [466] was used. Due to potential circularity, the text mining STRING network was not included in prior benchmarks, but we included it here to boost our performance. 256D network embeddings were generated using networks of the seven STRING edge types. The latest version of FunFams, v4.3, generated in January 2020, so were used. *S. pombe* protein sequences were searched against the v4.2 and v4.3 FunFams using hmmsearch. GO terms associated with FunFams were downloaded in February 2020, and all terms, except those with NAS, ND, TAS and IEA, were included, as well as UniProtKB-kw IEA terms. CAFA provided a frozen version of the Gene Ontology for every team to use (October 7, 2019 release). Up-to-date *S. pombe* GO annotations were downloaded from PomBase on January 14, 2020.

We experimented with many different combinations of features and submitted the three models that had the best AUPR from cross-validation. All three of the models were trained on the network embeddings. Additionally, v4.3 FunFams were used in model 1 and

v4.2 FunFams in model 2. Inclusion thresholds were used for v4.2 FunFam HMMs, whereas, a threshold of $E < 10^{-4}$ was used for v4.3 FunFam HMMs to mitigate any potential over-splitting that may have occurred whilst generating these FunFams. E-values were $-\log_{10}$-transformed to be used as FunFam features.

Random forest predictions were post-processed. Parents of predicted terms were added with the same probability as the predicted term. GO taxon constraints were applied to remove terms that never occur in *S. pombe*. Predictions were filtered to remove any duplicates and the prediction with the highest probability was kept. Predictions were submitted to CAFA 4 under the team name 'OrengoFunFamLab2'.

5.2.3 Data analysis

5.2.3.1 Clustering

Growth phenotype data was clustered [467] and heatmaps with dendrograms were plotted. Euclidean distance matrices were calculated [468] ('euclidean' metric in `scipy.spatial.distance.pdist` [469]). The distance matrix was clustered using the unweighted pair group method using arithmetic averages (UPGMA) algorithm [470] ('average' method in `scipy.cluster.hierarchy.linkage` [469]). UPGMA performs bottom-up hierarchical agglomerative clustering to form a dendrogram by iteratively merging the two most-similar clusters. Distances between clusters are calculated as the mean distance between all pairwise combinations of items between each cluster. By considering all items in clusters, UPGMA is more robust to outliers than single-linkage clustering, which only considers the nearest pair of items. Heatmaps were plotted using `seaborn.clustermap`.

5.2.3.2 Code

Julia 1.1-1.4 was used for most of this work. Julia packages used were: DataFrames v0.19.1, DataFramesMeta v0.5.0, DecisionTree v0.10.1, Distances v0.8.0, GLM v1.1.1, HypothesisTests v0.8.0, MLBase v0.8.0, MLDataUtils v0.5.0, MultipleTesting v0.4.1, OBOParse v0.0.1, PlotlyJS v0.13.1, Plots v0.27.1, StatsBase v0.31.0, StatsPlots v0.10.2, and UMAP v0.1.4. Custom code is available as a Julia package PombeAgeingGenes.jl on GitHub (https://github.com/harryscholes/PombeAgeingGenes.jl). Python 3.7 was used to plot clustermaps of matrices using numpy v1.15.4, pandas v0.23.4, matplotlib v3.0.2 and seaborn v0.9.0.

5.3 Results

5.3.1 Known functions of fission yeast proteins

Overarchingly, this project aims to accurately predict functions of *S. pombe* genes, so that experimentalists can be informed about which targeted functional validations to perform, so as to increase the total number of genes with at least one 'experimental' annotation, and to increase the total number of 'experimental' annotations. 4,523 (88%) of *S. pombe*'s 5,137 protein-coding genes have at least one 'experimental' annotation in any of the three ontologies—biological process, molecular function and cellular component (Fig. 5.1). However, if only biological process and molecular function terms are not considered, only 1,963 (38%) genes have at least one 'experimental' annotation. This is because it is relatively easy to determine the subcellular location of proteins in high-throughput screens, such as automated fluorescence microscopy.

For the biological process and molecular function ontologies separately, 4,190 (82%) and 3,310 (64%) of genes have at least one high-quality annotation in the 'experimental' or 'curated' classes of GO evidence codes. In the cellular component ontology, 4,449 (87%) of genes have at least one 'experimental' annotation, of which 2,790 (63%) are from high-throughput studies with evidence codes HTP, HDA, HMP, HGI or HEP, which may represent a significant bias in the functional distribution of *S. pombe* gene annotations [428]. Annotations in the 'bad' class should not be trusted. Together with genes that have no annotations, there is an opportunity to assign functions from the biological process, molecular function and cellular component ontologies to 845 (16%), 1,439 (28%) and 231 (5%) genes, respectively.

5.3.2 Colony size phenomics of fission yeast gene deletion mutants grown in a panel of conditions

The growth phenotype data was processed as described in Section 5.2.1. The entire data set contained 2,832,384 colonies. 2,389,858 colonies remained after 'grid' and 'empty' colonies were removed (Fig. 5.2). 2,256,475 colonies remained after poor-quality colonies were removed. Some conditions were supplemented with a red food dye called phloxine B that is used in yeast cell-viability assays [458]. The polar dye is excluded from healthy cells with intact membranes, but when membrane integrity is lost in dead cells, the dye is able to accumulate in cells and can be detected easily as red-stained colonies. In this way, dead cells can be distinguished from cells that are alive, but are unable to divide [471]. Colonies

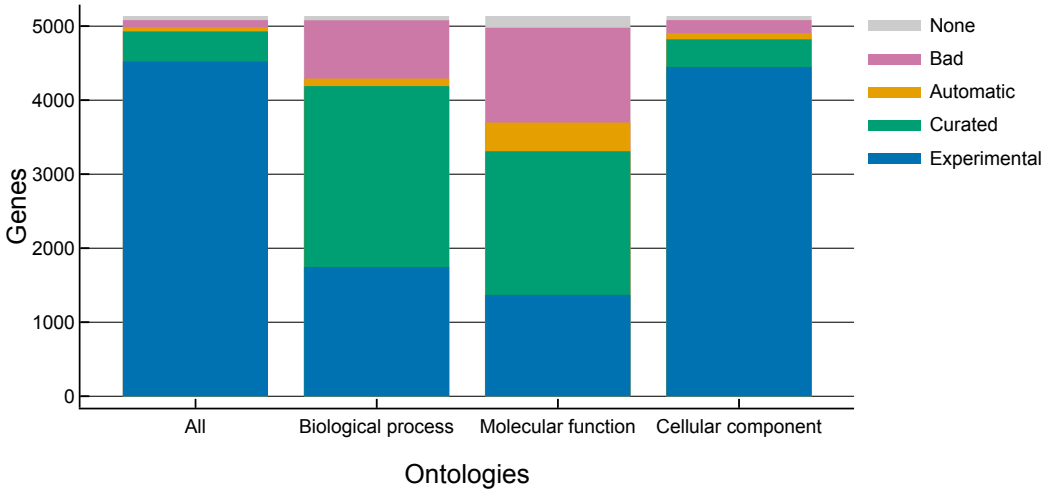


Figure 5.1: **Assessing the quality of GO term annotations in *S. pombe* genes.** For each ontology—biological process, molecular function and cellular component—together (All) and separately, *S. pombe* genes are assigned to one of five classes—experimental, curated, automatic, bad and none—in that order of preference.

that were grown in conditions that were, or were not, supplemented with phloxine B were pooled because phloxine B does not interfere with growth [472].

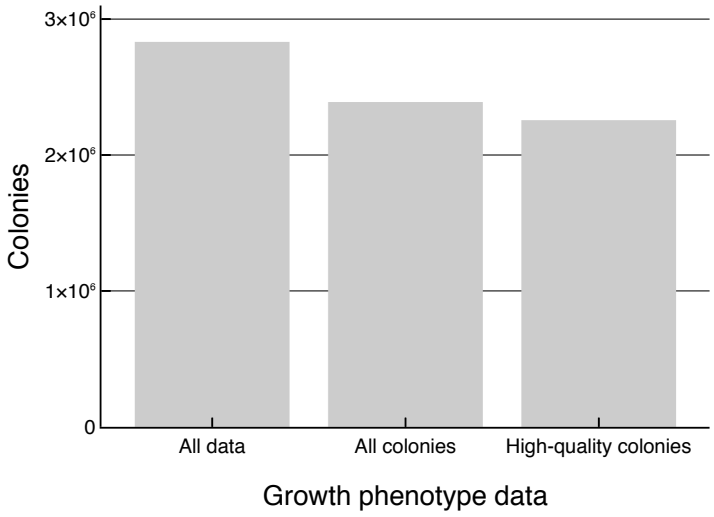


Figure 5.2: **Processing colonies in growth phenotype data.** 'All data': all colony size measurements in the data set, 'All colonies': colonies remaining after 'grid' and 'empty' colonies were removed, 'High-quality colonies': colonies remaining after poor-quality colonies were removed.

5.3.3 Estimating the reliability of colony sizes

For each strain-condition pair, the variance in colony size was calculated (Fig. 5.3). Variances are distributed exponentially, with a heavy positive skew. Generally, strain-condition pairs have low variance, with mean 0.0733 and 95^{th} percentile 0.287.

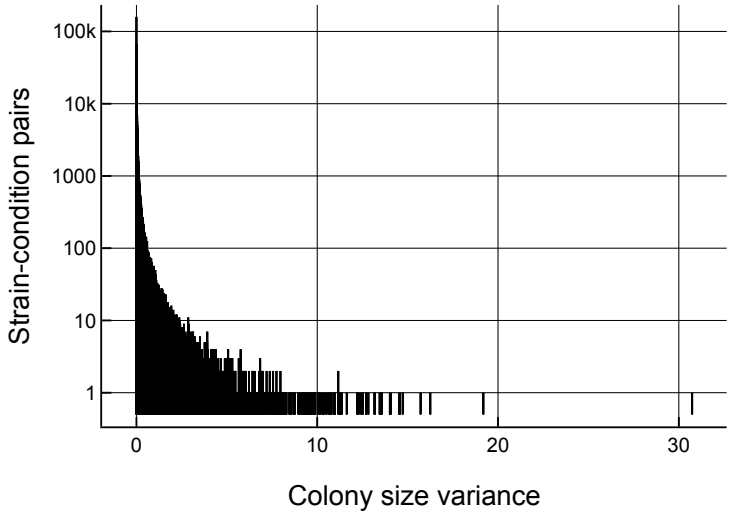


Figure 5.3: **Distribution of colony size variance across for all strain-condition pairs.** Variance in colony size for strain-condition pairs with > 1 repeat were calculated.

We assessed the reliability of strain colony sizes from multiple repeats in the same condition. By reliability, we mean how similar are the sizes of pairs of colonies from the same strain grown in the same condition. We chose the condition 'YES_SDS_0.04percent' because it had the most pairs of repeats, except for 'YES_32' and 'EMM_32'. Colony sizes from the $1{,}055{,}687$ pairs of repeats for particular strains agreed poorly (Fig. 5.4). These pairs of colony sizes had a Pearson correlation coefficient of $r = 0.215$. We also fitted a linear regression model to the pairs of colony sizes and found that this line deviates significantly from the ideal $y = x$ line—corresponding to perfect reliability—with a coefficient of determination $R^2 = 0.036$.

We observed that colonies of *S. pombe* gene deletion mutants displayed a high false negative rate, i.e. a growth phenotype was not observed, despite the knocked out gene being affected by the growth condition. In other words, strains would sometimes show a phenotype in one repeat, but inexplicably would not show any phenotype in other repeats.

Each strain-condition pair was then represented by the colony whose size had the largest effect size by (Eq. (5.1)). From now on, we refer to these values as the 'colony

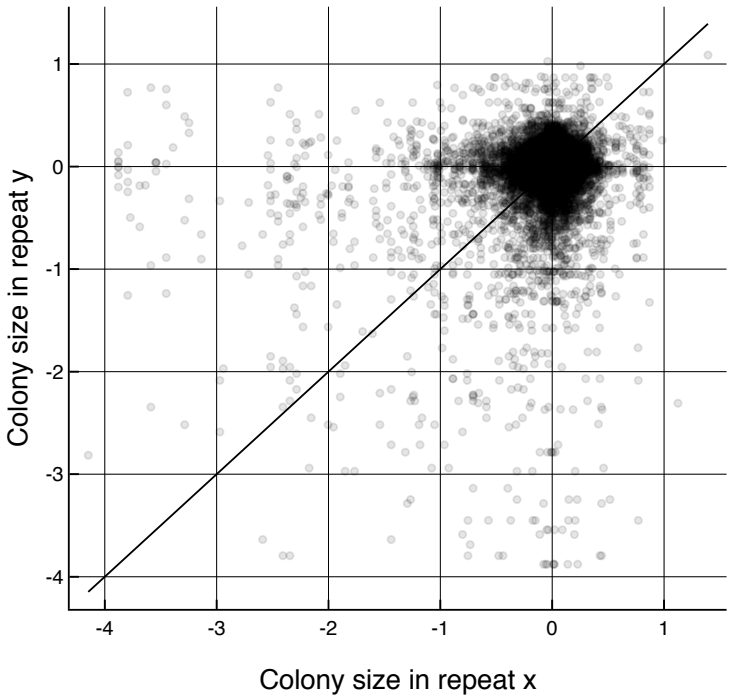


Figure 5.4: **Reliability of colony sizes.** The condition 'YES_SDS_0.04percent' was used. $1{,}055{,}687$ pairs of repeats for particular strains exists. $20{,}000$ pairs were randomly sampled and plotted.

size' for each strain-condition pair. Colony sizes across all strains and conditions follow a normal distribution with $\mathcal{N}(\mu = -0.046, \sigma^2 = 0.281)$ (Fig. 5.5). Mean colony size of -0.046 shows that strains tend not to have strong phenotypes in conditions, with a small trend towards being less fit. The variance of the fitted normal distribution agrees well with the empirical mean of colony size variance of each strain-condition pair (Fig. 5.3). Assuming normality, a standard deviation of 0.530 means that 95% of colonies will be within

$$\mu \pm 2\sigma = -0.046 \pm 0.530 = -0.576 < x < 0.484.$$

So taking the inverse logarithm results in 95% of corrected colony sizes between

$$2^{-0.576} < x < 2^{0.484} = 0.671 < x < 1.399.$$

On closer inspection, the distribution is bimodal about 0, due to the way that we calculated point estimates of colony sizes using $X[\text{argmax}(\text{abs}(X))]$. As the most extreme

colony size, x, tends to zero, the probability, $P(x)$, also tends to zero,

$$\lim_{x \to 0} P(x) \to 0$$

unless there is absolutely no phenotype, or no biological or technical error.

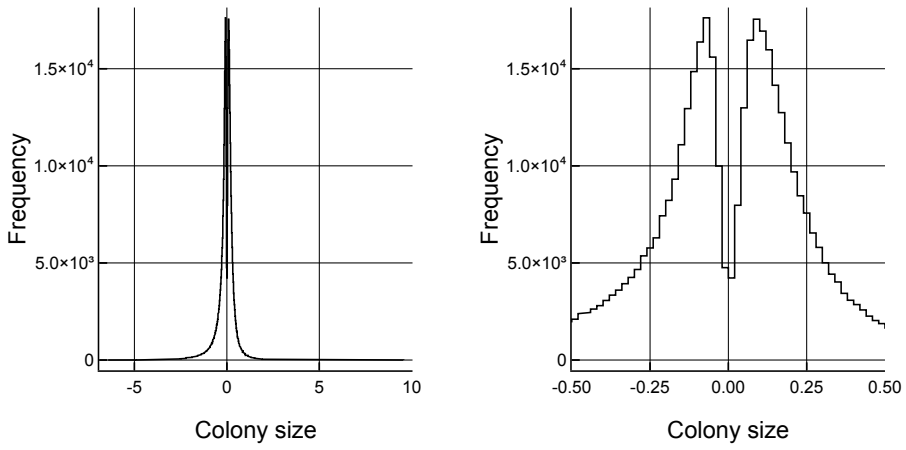


Figure 5.5: **Distribution of colony sizes from all strains and conditions.** Colony sizes for all colonies (left) and $-0.5 \leq x \leq 0.5$ (right) are shown. Colony sizes were corrected using the reference surface from the grid of wild type colonies on each plate, normalised to the strain's growth in control conditions, and $\log_2$-transformed. Point estimates of the colony size distribution for each strain-condition pair, X, was calculated by $X[\text{argmax}(\text{abs}(X))]$.

5.3.4 Identifying conditions that elicit strong phenotypes

Conditions that elicit strong phenotypes were identified by calculating the mean colony size and variance per condition (Fig. 5.6). In this context, strong phenotypes were defined as having mean colony size > 0.2 or < -0.2, or variance > 1.0. All of the three 10 mM EGTA conditions had high variance > 1.0, along with EMM C (ciclopirox olamine, an antifungal drug) 2 ug/ml. EGTA is a strong chelating agent that binds positively charged metallic magnesium and calcium ions. Conditions with small mean colony sizes were YES_KCl_0.5M_SDS_0.04percent and YES_Diamide_2mM. 0.6 M KCl decreases permeability of the *S. pombe* cell membrane and increases its resistance to certain drugs [473], but 0.5 to 2.0 M KCl interrupts logarithmic growth in *S. cerevisiae* and is mutagenic [474]. YES_KCl_0.5M_SDS_0.04percent had a mean colony size of -0.270, whereas, YES_KCl_1M_SDS_0.04percent on average were not affected as strongly and had mean colony size of -0.035. All other conditions with 0.04% SDS did not have as strong effect sizes. Either, this is an outlier, or 0.05 M KCl does not protect the cell membrane

against detergent or acts synergistically to solubilise the membrane. Diamide is an oxidising agent that is toxic to *S. pombe* at 3 mM, and in the YES_Diamide_2mM condition, the mean colony size was -0.208.

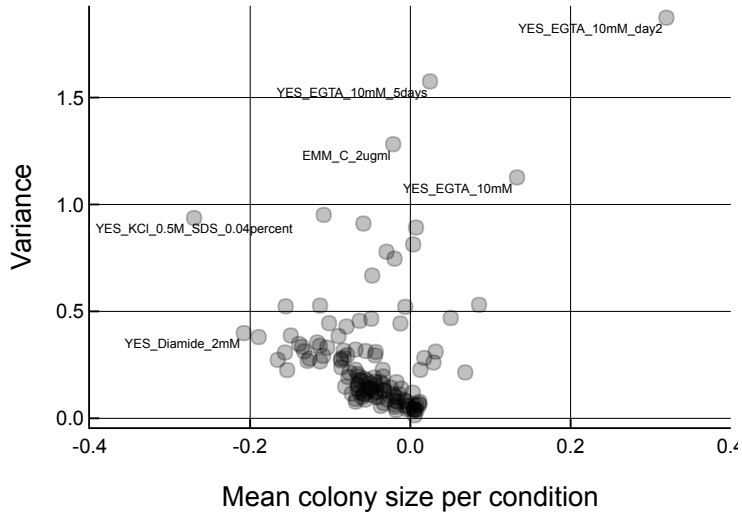


Figure 5.6: **Mean colony size per condition across all strains, against the variance.** Conditions that elicit strong phenotypes, with mean colony size > 0.2 or < -0.2, or variance > 1.0, were labelled.

5.4 Clustering strains and conditions according to phenotypic patterns

To identify groups of strains and conditions that had similar patterns of colony growth across all strains or all conditions, we clustered a [strains $\times$ conditions] matrix of colony sizes using Ward's method. The clustered heatmap is divided into two main clusters, where there smaller cluster at the bottom of the heatmap are sensitive to many conditions (Fig. 5.7). The larger cluster at the top of the heatmap are not affected by many conditions, or are resistant to certain blocks of conditions. Many of the colonies in the lower cluster are also resistant to these conditions though, which is an interesting finding, as it shows that the wild type grows worse in these conditions than the gene deletion mutants. The choice of media appears to affect the growth of colonies across the different strains, as shown by EMM-based conditions clustering into a block of conditions (pink), sandwhiched between YES-based conditions (green).

We also calculated correlation matrices for conditions (Fig. 5.8) and strains (Fig. 5.9) and clustered these using UPGMA. The correlation matrix for conditions (Fig. 5.8) is the

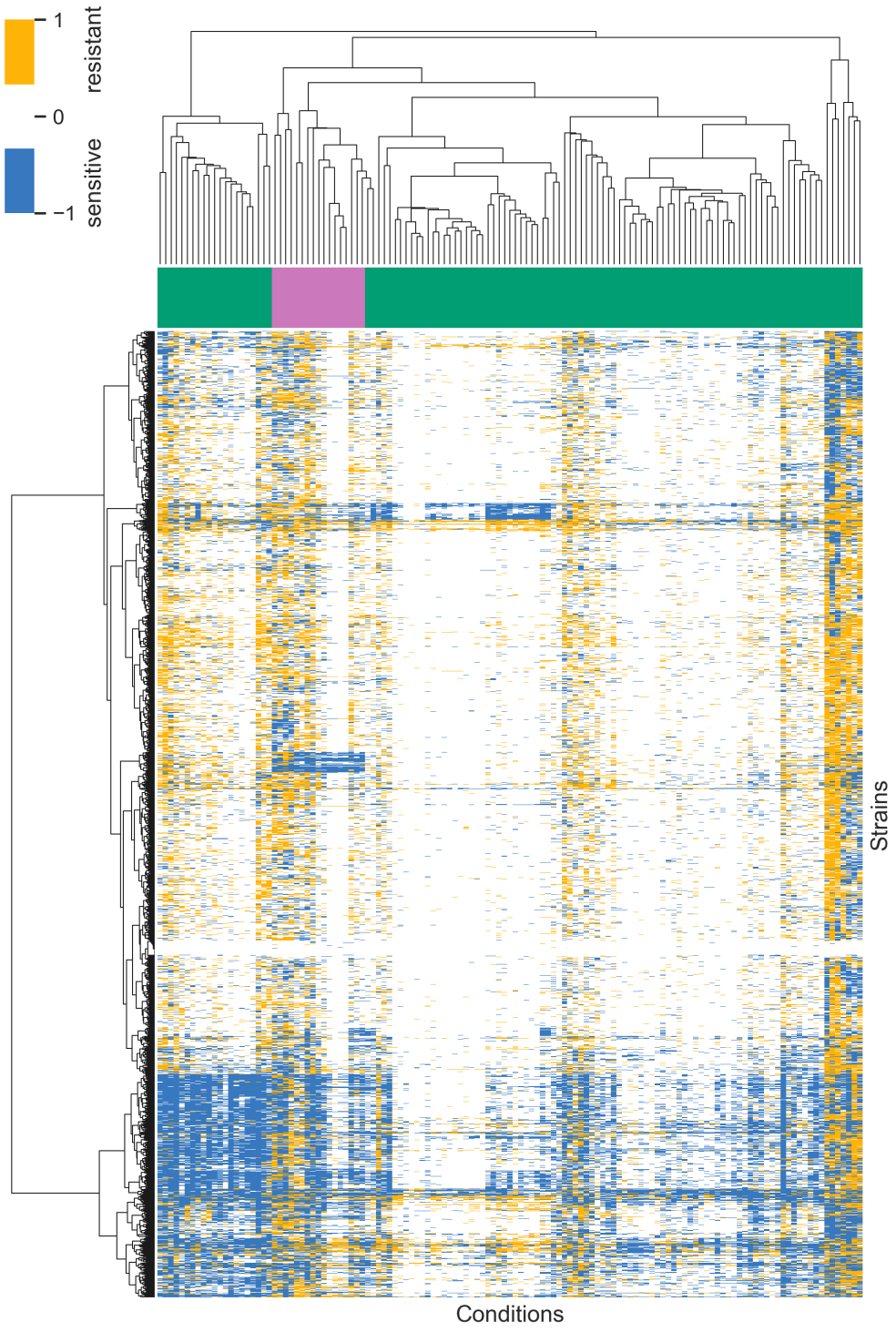


Figure 5.7: **Sensitivity of strains in conditions.** A threshold of 1.25-fold growth difference was set for resistant strains, where $\log_2(1.25) = 0.32$. Resistant colonies (yellow) have $\log_2$-transformed sizes > 0.32, and sensitive colonies (blue) have $\log_2$-transformed sizes < -0.32. The Euclidean distance matrix was clustered using Ward's method. Columns are coloured by media type, where green denotes YES media and pink denotes EMM media.

easiest of the three clustered heatmap figures to interpret because it is the smallest. Conditions based on EMM media cluster into two clades, that are not contaminated with any YES-based conditions. Furthermore, 11 out of the 17 EMM conditions can be easily separated from the remaining conditions in the first UMAP embedding dimension [475] (data not shown). UMAP is a dimensionality reduction method similar to, but more powerful than, principal component analysis. Most YES glucose conditions form a clade at the bottom right corner of (Fig. 5.8), with some contamination from other sugars (galactose, xilose and mannitol) and alcohols (glycerol and ethanol), which are metabolised similarly. Interestingly, although tea tree oil is a potent antifungal [476], the two tea tree conditions also clustered into this clade. This is likely because all of these conditions result in small colonies.

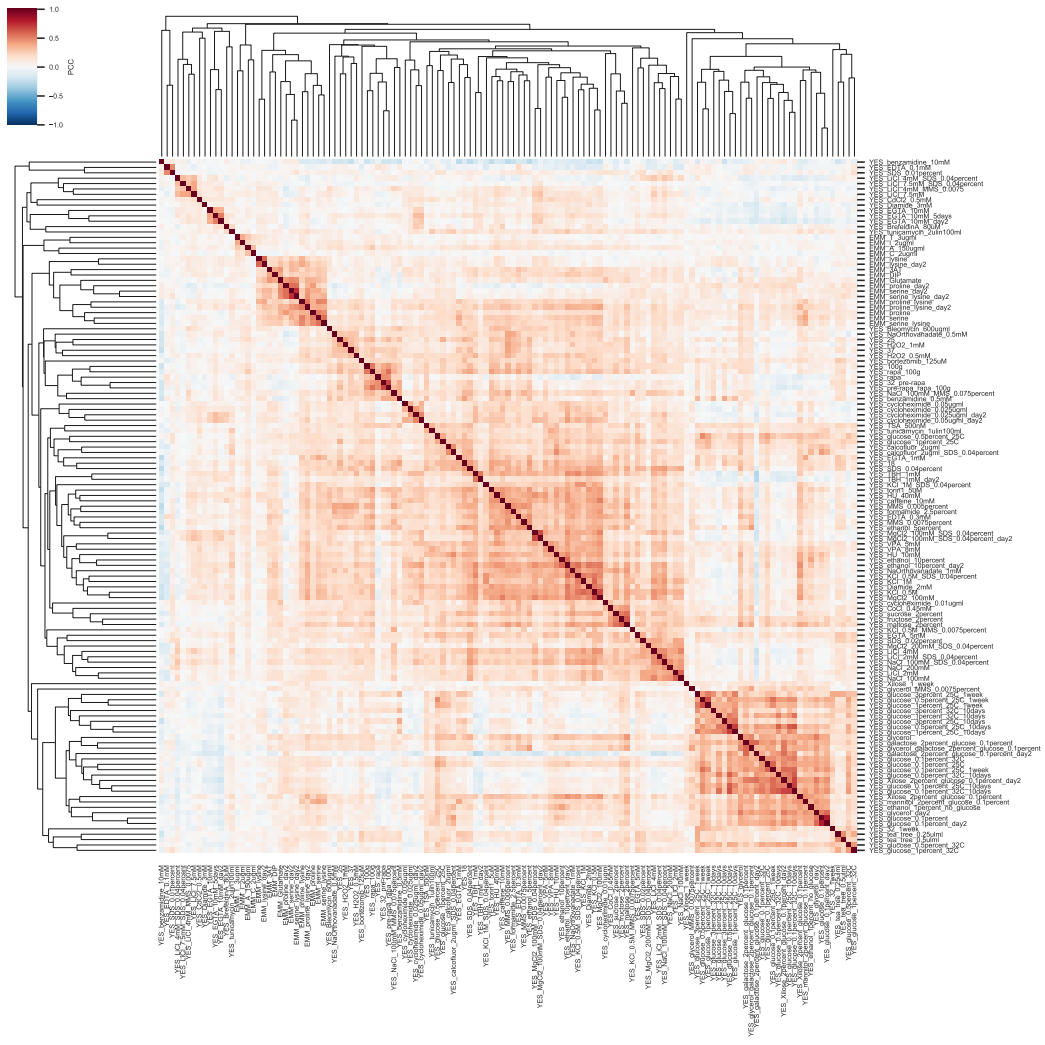


Figure 5.8: **Correlation of conditions.** The Pearson correlation coefficient (PCC) matrix of conditions across all strains was clustered using UPGMA.

The study that identified tea tree as lethal to *S. pombe* found the minimum inhibitory concentration to be 0.5% v/v, but the conditions that we used were 0.25 µl/ml and 0.50 µl/ml, i.e. 5% and 10% of the minimum inhibitory concentration. Low concentrations of antibiotics are known to act as environmental signals for bacteria that cause multifarious cellular changes [477]. Although the mechanism of tea tree's antifungal properties is not known, it is thought to affect the permeability of membranes [478]. However, a tight clade was formed with the two tea tree conditions, YES_glucose_0.5percent_32C and YES_glucose_1percent_32C, so it is also possible that tea tree had a minimal effect on cells at such low concentrations.

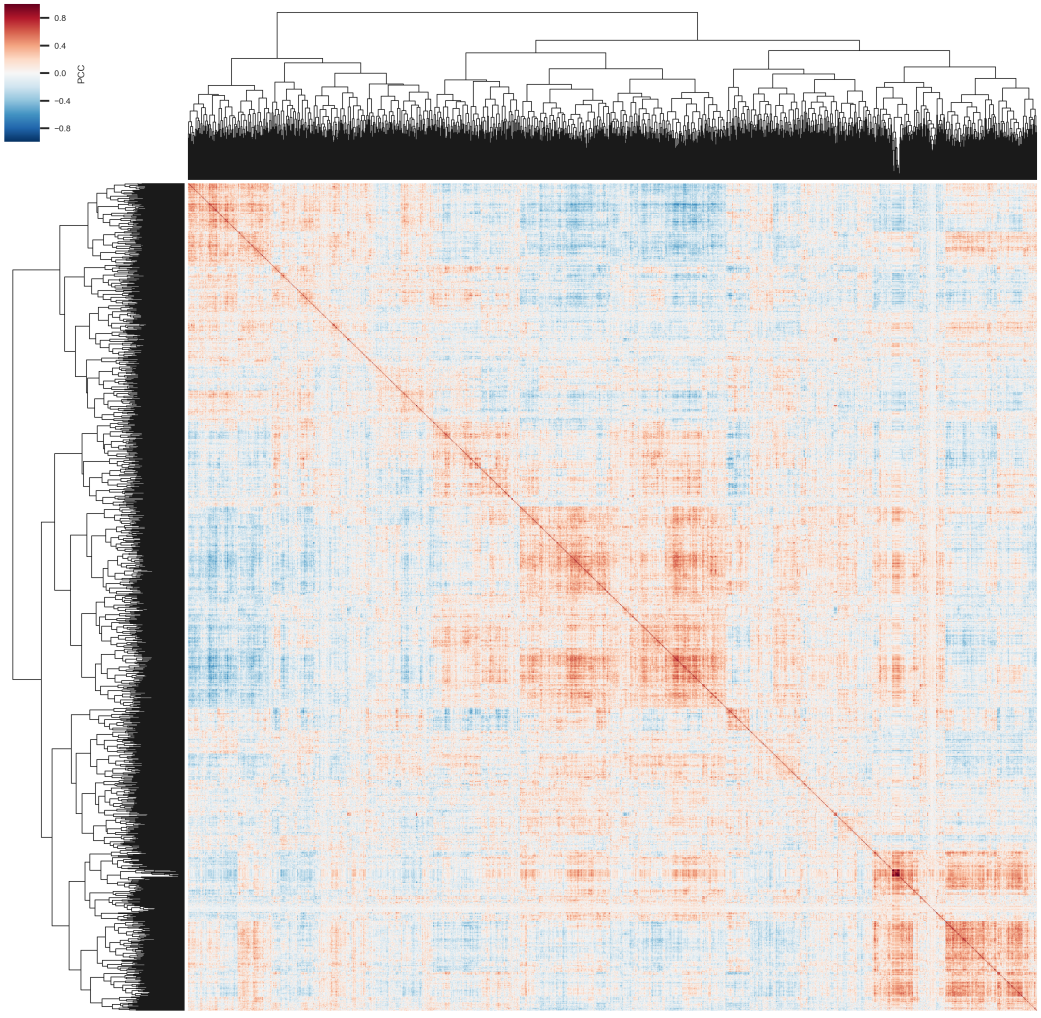


Figure 5.9: **Correlation of strains.** The Pearson correlation coefficient (PCC) matrix of strains across all conditions was clustered using UPGMA.

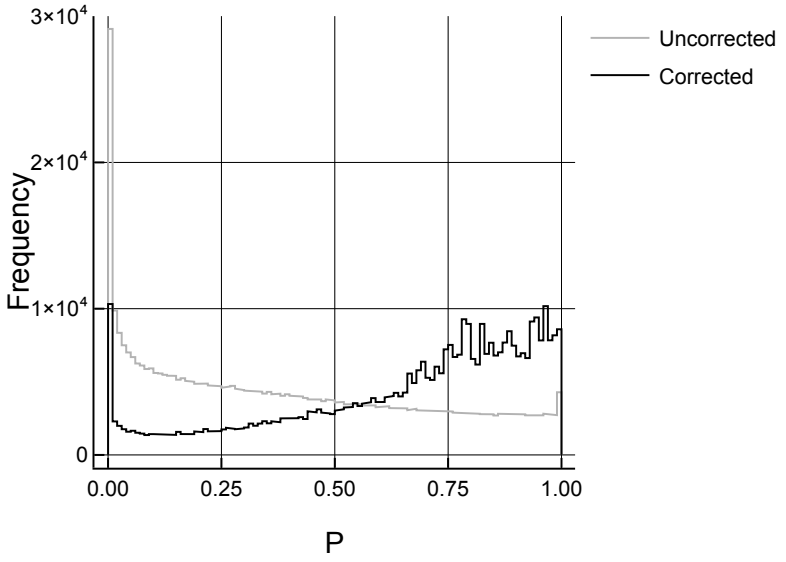
5.5 Significance testing to identify significant phenotypes

Strains whose growth were affected in particular conditions were identified using null hypothesis significance testing. Absence of knocked out genes may affect growth, regardless of the condition, and in particular may have a negative affect on growth, as suggested by the mean colony size of -0.046 that we measured across all strains and conditions. To account for these effects, the null distribution for each strain is the distribution of its colony sizes in plain media at 32°C: 'YES_32' for conditions based on YES media, or 'EMM_32' for EMM media. For each strain and condition, the colony size distribution was compared to its corresponding null distribution using the two-sample unequal variance t-test (also known as Welch's test). The null hypothesis was that there is no difference between how a strain grows in a condition and its corresponding control condition. We used the two-sample unequal variance t-test because colony sizes of strains grown in the control condition follow a distribution, and this distribution might not have the same variance as non-control conditions.

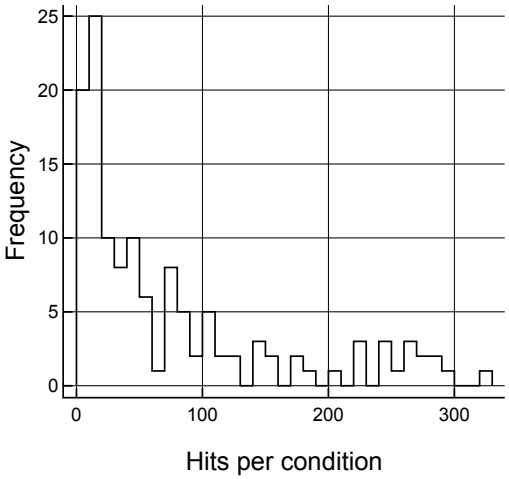
Conditions are independent of other conditions. We considered each condition to be a separate experiment, so applied multiple testing correction to P values from each condition separately [479]. We believe that it makes more biological sense to consider each condition a separate experiment and not each strain as a separate experiment. In total, $10{,}318$ hits were called at the $Q < 0.01$ significance level (Fig. 5.10a). 128 out of the 131 conditions had at least one hit, with a median of 43 hits per condition (Fig. 5.10b). Other than the two control conditions, the only other condition that had no hits was 'YES_Diamide_3mM' because only one repeat was performed for this condition because it was too toxic. $3{,}058$ strains out of $3{,}510$ strains that we screened had at least one hit, with a median of 2 and a mean of 3 hits per strain (Fig. 5.10c). Eight strains had hits in more than 20 conditions (Table 5.2), consisting of transcription factors, kinases (protein kinase gsk3 and serine/threonine-protein kinase cds1) and a critical metabolic enzyme (fructose-1,6-bisphosphatase).

5.5.1 Assigning fission yeast proteins to FunFams

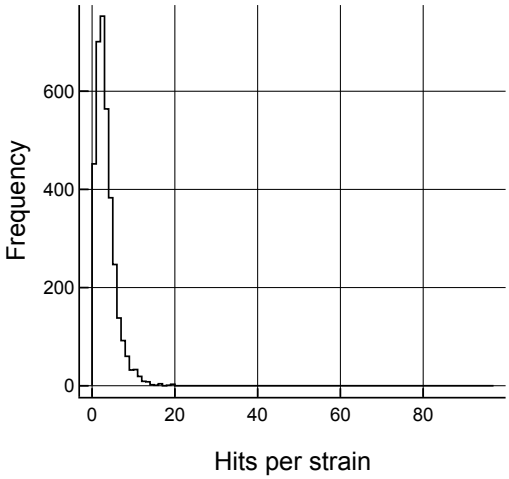
S. pombe proteins were mapped to FunFams, to encode homolog information when using machine learning to predict GO annotations. Sequences of the $5{,}137$ proteins encoded in *S. pombe*'s genome were scanned against HMMs from each of the $68{,}065$ FunFams. At a threshold of E $< 10^{-3}$, 3,319 (64%) proteins had $149{,}099$ hits to $23{,}900$ FunFams



(a) **Testing whether strain growth is significantly affected by conditions.**



(b) **Number of hits per condition.**



(c) **Number of hits per strain.**

Figure 5.10: **Significance testing. a.** Testing whether strain growth is significantly affected by conditions. The null hypothesis was that there is no difference between how a strain grows in a condition and its corresponding control condition—'YES_32' for conditions based on YES media, or 'EMM_32' for EMM media. Two-sample unequal variance t-tests were performed for colony size distributions and their corresponding null distributions. **b.** Number of hits per condition. Two-sample unequal variance t-tests were performed for colony size distributions and their corresponding null distributions. P values were corrected for multiple testing using the Benjamini-Hochberg method. Hits were called at the $Q = 0.01$ significance level. **c.** Number of hits per strain. Two-sample unequal variance t-tests were performed for colony size distributions and their corresponding null distributions. P values were corrected for multiple testing using the Benjamini-Hochberg method. Hits were called at the $Q = 0.01$ significance level.

Table 5.2: **Number of hits per strain.**

Gene ID	Symbol	Gene long name	No. hits
SPBC106.10	pka1	cAMP-dependent protein kinase	97
SPBC725.11c	hap2	Transcriptional activator hap2	78
SPBC1105.14	rsv2	Zinc finger protein rsv2	70
SPBC29B5.01	atf1	Transcription factor atf1	64
SPBC1198.14c	fbp1	Fructose-1,6-bisphosphatase	54
SPAC1687.15	gsk3	Protein kinase gsk3	50
SPCC18B5.11c	cds1	Serine/threonine-protein kinase cds1	36
Wild type	-	-	30

(35%). 10,136 of these FunFams (42%) were hit by only one protein (Fig. 5.11). The median number of hits per FunFam was 2 and the mean was 5.35. The maximum number of hits per FunFam was 126 proteins for the WD40 repeat-containing serine/threonine kinase FunFam from the YVTN repeat-like/Quinoprotein amine dehydrogenase superfamily (2.130.10.10/FF/102735).

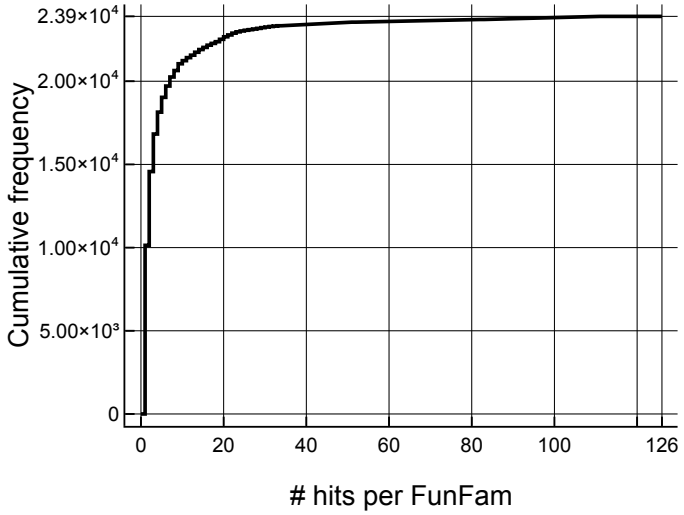


Figure 5.11: **Number of hits per FunFam for *S. pombe* proteins.** Proteins were scanned against the CATH v4.2 FunFam HMM library using an E-value threshold of $E < 10^{-3}$. Only FunFams with at least one hit are shown.

We associated GO terms to FunFams by identifying all GO terms that are annotated to proteins in each FunFam (Fig. 5.12). FunFams were associated with GO Slim terms with a median of 1,619 across the 53 *S. pombe* GO Slim terms. 'Vesicle mediated transport' (GO:0016192) was associated with the maximum number of 11,279 FunFams. Non-parametric testing, using the number of descendants each term has as the null distribution,

indicates that 'vesicle mediated transport' has significantly more descendant terms than expected, $P = 0.011$, which may go some way to explain why 17% of FunFams contain sequences that are annotated with terms related to 'vesicle mediated transport'.

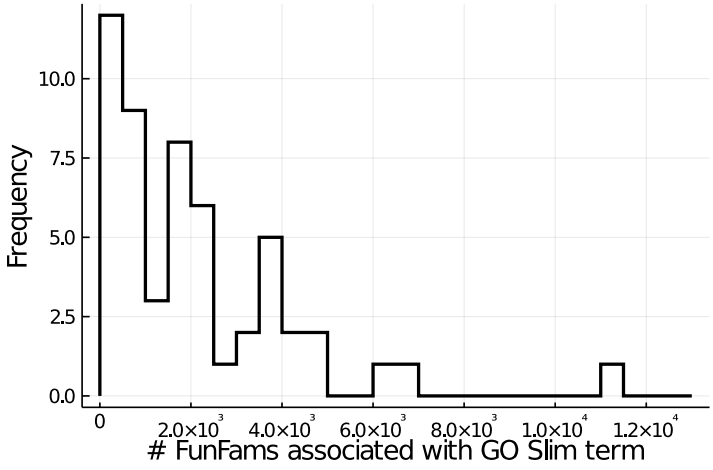


Figure 5.12: **Numbers of FunFams associated with GO Slim terms.** GO terms were associated to FunFams by identifying all GO terms that are annotated to proteins in each FunFam. All annotations were included, except those with NAS, ND, TAS or IEA evidence codes, but UniProtKB-kw IEA curated terms were included. Ancestor terms that have 'is_a', 'has_part', 'part_of' and 'regulates' relationships were also included.

5.5.2 Benchmarking machine learning protein function prediction models

GO Slim terms for *S. pombe* were predicted using RFs and the one-vs-rest classification strategy. Different combinations of features were used to predict GO Slim terms and their performance was evaluated (Fig. 5.13). The prediction error was estimated using five independent repeats of 5-fold cross-validation. We present the results for models trained using one type of data first and then go on to discuss models trained using a combination of different data.

Chapter 4 reported how deepNF [66]—a state of the art gene function prediction method—was applied to *S. pombe* networks to predict GO terms. Here, we used the best 256D network embeddings from Chapter 4 as features to predict GO Slim terms. Alone, network embeddings were the best set of features (AUPR $= 0.583$; yellow curve). The next best features were FunFam homology data (AUPR $= 0.424$; orange curve). Third best were the yeast growth phenotype data (AUPR $= 0.126$; light blue curve). At first glance, this performance may not appear so good, however, the classifier is many times better than random. *S. pombe*'s GO Slim annotations have $|\mathrm{P}| = 7{,}646$ and $|\mathrm{N}| = 224{,}706$, so a

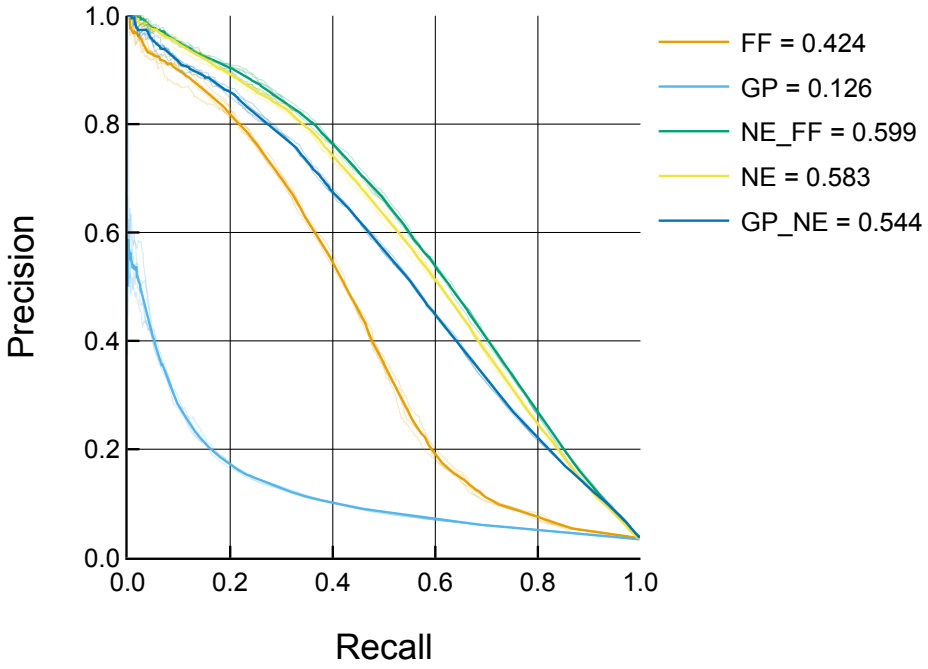


Figure 5.13: **Precision-recall curves for predicting GO Slim terms.** The 53 GO Slim terms were predicted using RFs and a one-vs-rest classification strategy. The prediction error was estimated using five independent repeats of 5-fold cross-validation. Precision-recall curves are plotted for each repeat (thin translucent curves) as well as a micro-averaged curve (thick solid curves). Numbers in legend correspond to the micro-averaged AUPR. Legend abbreviations: GP, growth phenotypes; NE, network embeddings; FF, FunFam homology data.

random classifier would have AUPR $= 0.0329$. Therefore, these growth phenotype data are able to predict GO Slim terms 3.8 times better than a random classifier, but, compared to network embeddings and FunFam data, the growth phenotypes are poorly predictive of GO Slim annotations.

Two combinations of features were tested. A combination of network embeddings and FunFam data produces a performance 3% higher than the network embeddings alone (AUPR $= 0.599$; green curve). Due to the way that we performed the cross-validation, this is a genuine—albeit modest—increase in performance. Another combination of growth phenotypes and network embeddings produces a performance 7% lower than the network embeddings alone (AUPR $= 0.544$; dark blue curve).

5.5.3 Annotating fission yeast proteins with new functions

GO terms that are annotated to between 50 to $1{,}000$ *S. pombe* proteins were predicted with AUPR $= 0.569$ (Fig. 5.14). $2{,}390{,}915$ annotations were predicted, of which $97{,}017$ were al-

ready known and in the target set. After removing known annotations, with experimental or curated evidence codes, and 2,167,057 predictions with probability $P(\text{annotation}) < 0.1$, 126,841 predictions remained for 534 GO terms in 4,456 proteins. Only 2,628 (2%) of these predictions were present in the *S. pombe* GO annotations with IEA evidence codes, and 7,710 (6%) with IEA, NAS or ND evidence codes. We do not predict any functions for proteins that had no annotations (regardless of evidence codes), and we also do not predict functions of proteins that had no experimental or curated annotations (i.e. only IEA, NAS or ND evidence codes). We do, however, predict 117,654 new functions for proteins that previously had experimentally validated functions.

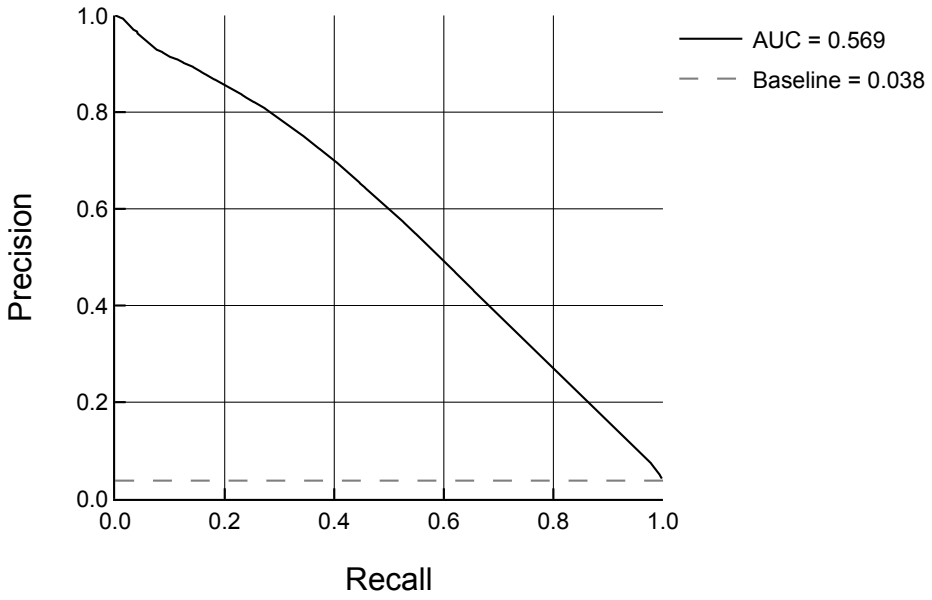


Figure 5.14: **Performance of predicting functions of *S. pombe* proteins.** The model was trained on network embeddings and FunFams. GO terms that are annotated to between 50 to 1,000 proteins are included. A precision-recall curve is plotted. Numbers in legend correspond to the micro-averaged AUPR.

692 *S. pombe* proteins have unknown function, of which 409 have orthologues in other organisms and 145 of these are conserved in vertebrates—the so called 'priority unstudied genes' defined by PomBase. (NB some 'unknown function' and 'priority unstudied' proteins have automatically assigned functions with the IEA evidence code.) We analysed the functions that were predicted for proteins conserved in vertebrates and the priority unstudied proteins (accessed on June 10, 2020). For the conserved proteins, we predicted 4,285 functions for 397 GO terms in 153 proteins (22%). 18 of these annotations were previously predicted with IEA evidence codes. For the priority unstudied proteins, we

predicted 1,865 functions for 350 GO terms in 64 (44%) proteins. 18 of these annotations were previously predicted with IEA evidence codes. 133 of the 145 priority unstudied proteins are in the Bioneer gene deletion mutant library, and 60 out of the 64 priority unstudied proteins that we predicted functions for are in the Bioneer collection.

5.5.4 CAFA 4

We entered CAFA 4 with predictions for fission yeast GO terms, under the team name 'OrengoFunFamLab2'. We experimented with many different combinations of features and submitted the three models that had the best AUPR from cross-validation (Fig. 5.15). Our three models were all trained on network embeddings. Additionally, v4.3 FunFams were used in model 1 and v4.2 FunFams were used in model 2. Our initial model—using network embeddings and v4.3 FunFams—produced AUPR $= 0.473$ (results not shown). After extensive experimentation (results not shown), we were able to to increase the AUPR of our final models considerably. Despite this, model performance appears to be asymptotic to AUPR ≈ 0.58.

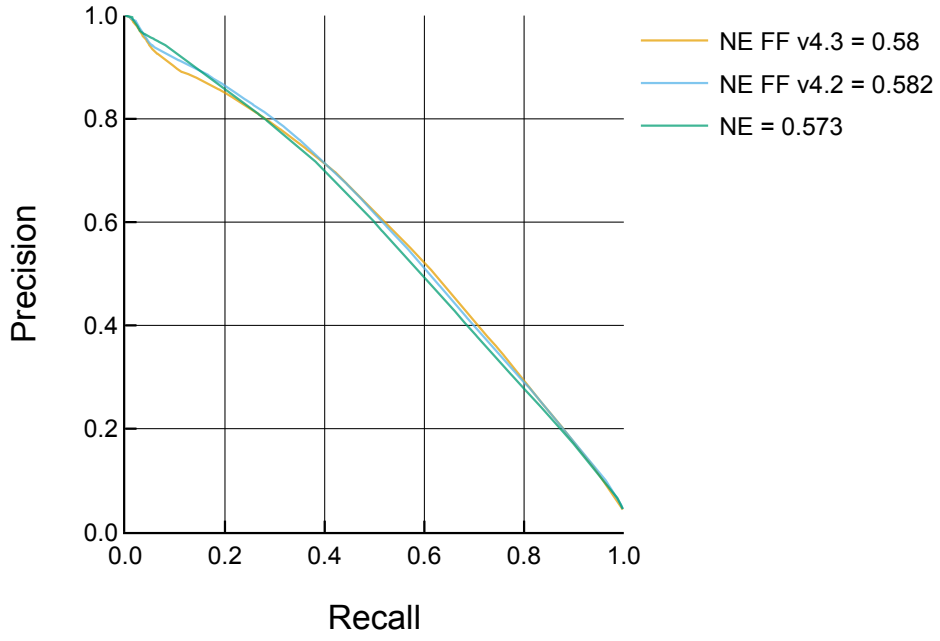


Figure 5.15: **Performance of three models submitted to CAFA 4.** Models were trained on different combinations of network embeddings (NE) and FunFams (FF). Precision-recall curves are plotted. Numbers in legend correspond to the micro-averaged AUPR.

We can understand how the various features contribute to the models by analysing the shape of precision-recall curves. Network embeddings produce high precision predic-

tions at low recall, whereas, FunFams increase the precision of predictions at high recall. If the goal is to assign functions to a set of proteins, without making any incorrect predictions, or missing any predictions, then it may be beneficial to use FunFam information, possibly in combination with network embeddings.

Though not directly relevant to this chapter, I also participated in the Orengo group's FunFam-based predictions for all 18 species, under OrengoFunFamLab team's 1, 3, 4 and 5. The preliminary results of CAFA 4 were presented at the ISMB conference in July 2020. For each ontoloy, the top 10 methods were presented, where each research group could only occur once in the top 10. The OrengoFunFamLab3 method came top in molecular function, third in biological process, and was not placed in the top 10 for cellular component. OrengoFunFamLab3 used CATH v4.3 and InterPro data to train an XGBoost classifier using a learning to rank strategy.

5.6 Discussion

5.6.1 Many factors may contribute to colony size phenomics being unreliable

Colonies of *S. pombe* gene deletion mutants displayed a high false negative rate for phenotypes in conditions and some strains had a high variance in colony sizes in particular conditions (Section 5.3.3). We assessed the variance in measuring colony sizes, due to technical errors associated with scanning plates, and found it to be very low. We normalised colony sizes to account for spatial biases, and the position of strains relative to each other, associated with plates. Therefore, false negative phenotypes appear to be genuine biological phenomena, which could have many contributing factors, including:

- viability of cells after thawing the gene deletion mutant collection,
- number of cells pinned on the plate to seed each colony,
- temperature and other environmental conditions of the laboratory during preparation of plates,
- temperature and other environmental conditions in the incubator during growth,
- nutrients and their concentrations within the agar plate.

The Bioneer collection contains gene deletion mutant strains of *S. pombe* for all non-essential genes. Genes may be non-essential because of genetic redundancy, or because the gene is not required in benign standard laboratory growth conditions. By stressing gene deletion mutants in a panel of growth conditions, we hoped to trigger condition-

dependent reduced fitness (slow growth) or condition-dependent essentiality (no growth) of the deleted genes. Genetic redundancy creates alternative routes for flux through metabolic and signalling pathways. *S. pombe* has not undergone whole-genome duplications [480], unlike *S. cerevisiae* [481], but instead its paralogs arose from small-scale duplications of chromosome regions. Whole-genome duplications give rise to paralogs with redundant functions, whereas, paralogs from small-scale duplications tend to be functionally divergent [482] or subfunctionalised [483, 484]. As such, it is reasonable to think that *S. pombe* would not exhibit much genetic redundancy—after all, carrying two genes that perform the same function is a fitness cost. However, we have previously shown that *S. pombe* rewires its gene expression and protein interaction networks after stress, to introduce redundancy and increase resilience to mutation [485, 486]. By subjecting *S. pombe* to stressful conditions, instead of eliciting phenotypes, we may have inadvertently triggered it to become more tolerant.

Data from multiple, independent repeats are often processed to obtain a point estimate that estimates the population distribution. A measure of central tendency, like the mean or median, is usually taken. If the false negative rate for phenotypes is caused by an underlying stochastic mechanism, or a mechanism that we perceive as stochastic, then central tendencies will obfuscate any true positives. For example, consider four colonies with sizes $[-1.0, -0.1, 0.0, 0.1]$, where -1.0 is a true positive and -0.1, 0.0 and 0.1 are false negatives. Here, the mean is -0.25 and the median is -0.05, which would both fail to capture the ground truth phenotype. To combat the high false negative error rate, we took as point estimates the maximum observed effect size from any repeat. That is, in our toy example above, the point estimate is -1.0, which successfully captures the ground truth phenotype. The downside of this approach is the possibility of increasing the false positive rate, which may arise if, for example, the grid normalisation failed to adequately normalise a plate's spatial biases. False positives are not as detrimental as false negatives. False positives can be filtered out by experimental screening or using evidence from the literature, but false negatives will not be tested because they would not be in the set of predicted functions.

5.6.2 FunFams were used to encode homology information in a novel way for machine learning

This is the first time that FunFam homology information has been used by us in a machine learning context. We encoded this information as a matrix of log-transformed HMM E-values. The resulting matrix is high-dimensional and sparse. HMM-based features have been used previously to predict protein function with machine learning [487–491], including logarithms of E-values [487]. High-dimensionality and sparsity are not ideal properties for machine learning, but we attempted to mitigate their negative effects using a novel training strategy that, to our knowledge, has never been used before. We trained models using the one-vs-rest strategy and only included FunFams that are associated with the GO term being predicted, thus reducing the dimensionality of the feature space (Fig. 5.12). When this work was conducted, deepNF was one of the best function prediction methods [66], so it was encouraging to see that the FunFams were able to improve on deepNF's performance.

All GO Slim terms are in the 'biological process' ontology. Network-based features were more predictive of GO Slim terms than FunFam-based features. At least two factors may contribute to this phenomenon. First, FunFams are groups of functionally pure proteins from different species, therefore they tend to be better at predicting GO terms from the 'molecular function' ontology, rather than the 'biological process' ontology [73, 74]. In CAFA 3, for example, the Orengo-FunFam team was ranked second place for predicting 'molecular function' terms and fourth place for 'biological process' terms. Therefore, this may explain only the modest performance improvements achieved when training models using FunFam and network embedding features. Second, *S.pombe* has been characterised extensively, so has comprehensive network data compiled across a large number of separate experiments. It could be that the quality of network data in *S. pombe* is very high and this effect would also be observed in popular model organisms, but not in less well-studied species.

The WD40 repeat-containing serine/threonine kinase FunFam that was hit by the most *S. pombe* proteins is large and contains 113,743 sequences from UniProt. The inclusion threshold—the HMM's trusted cutoff value—for this FunFam is 6.00, which is very low and suggests that either the FunFam multiple sequence alignment is poor, or this FunFam is affected by a known bug in CATH v4.2 FunFams. Sequences from a FunFam multiple

alignment are scanned against the corresponding FunFam HMM and the lowest bit score is used as the inclusion threshold. We recently discovered a problem that affects some FunFams, whereby short sequences, or subsequences, matched the HMMs with commensurately small bit scores. As such, sequences may be assigned to FunFams erroneously. Despite this, we can be reassured about the false positive rate because 95% of FunFams are hit by no more than 21 proteins.

5.6.3 Network data is powerful at predicting protein function

Network embeddings were the most predictive set of features for *S. pombe* protein function. We used deepNF to generate low-dimensional embeddings of proteins using information about their context across multiple networks [66]. deepNF is a highly competitive protein function prediction method. This is somewhat surprising, given that the method only uses network data. Until recently, network data had a bad reputation for being noisy and incomplete [492–495], but recent work suggests that networks are now more reliable [66, 116, 117, 496–498]. For example, GOLabeler [499] was the best method overall in CAFA 3 [74] (Zhu Lab team in CAFA 3), but NetGO [498], a model that adds network data to GOLabeler, was found to improve performance.

It has not escaped our attention that all of the work cited above use STRING [234, 466] as their sole source of network data. The authors state that STRING "aims to collect, score and integrate all publicly available sources of protein–protein interaction information, and to complement these with computational predictions" [466]. It is conceivable that the power of network data actually results from STRING's coverage, high-quality curation and accurate predictions, because it is known that integrating information from independent sources improves predictions [500].

The growth phenotype data were acquired using a high-throughput plate colony size assay. These data constitute a single screen that has associated biological and technical error rates [501], as opposed to STRING [466], which is, in essence, a meta-analysis of all known protein-protein interaction information, with far lower error rates.

It may appear surprising that, when combined with the network embeddings, the growth phenotypes cause a reduction in performance, but this can happen for the following reasons. Firstly, each tree in the forest is trained on random subsets of features from the training data that are selected from 131 growth phenotypes and 221 network embedding dimensions, so the growth phenotypes account for 37% of features. Given that we

know growth phenotypes are much less predictive than network embeddings, it is almost surprising that the reduction in performance is not larger than 7%, relative to network embeddings alone. Secondly, growing trees uses a greedy algorithm, so the associated error may not be equal to the global minimum, but rather may be an artefact of the heuristics used in the algorithm.

5.6.4 On CAFA and its value

Participating in CAFA 4 was a very valuable academic experience. Developing function prediction methods in isolation, without releasing predictions to the public, is futile. However, not every computational researcher or group is lucky enough to be able to collaborate with experimentalists that can validate their predictions. CAFA, the triennial evaluation of protein function prediction methods, was set up to provide a robust validation, without the need for explicit collaborations. Models are benchmarked on unseen data using a time-delayed evaluation to accumulate new experimentally-validated functions. This is in contrast to CASP [67], the community benchmark of protein structure prediction methods, which uses newly solved structures to evaluate model performance. In so doing, CAFA acts as a community benchmark of protein function prediction methods and captures the zeitgeist of the data, methods and models that are used to predict protein function.

Some interesting questions were raised during CAFA due to the evaluation strategy: Which models should be developed? To what extent, during development, should model choice be influenced by performance on benchmarks? How can overfitting on benchmarks be avoided, whilst still performing well on the evaluation data set? It is vital to not overfit models to benchmarks because the annotations in benchmarks are unlikely to be representative of the annotations that will accumulate to form the evaluation data set. Instead, models should be developed using general biological principles and our intuition [502]. In other words, if a model, that performs well in benchmarking, looks unlikely, it probably is. Here, we stuck to three biological principles that functions are: conserved through evolution (FunFams), encoded in how proteins interact (network embeddings), and functions have phenotypic consequences (growth phenotypes).

Here, we only predicted functions for one of the 18 model organism proteomes included in CAFA 4. The small number of proteins in the *S. pombe* proteome meant that evaluating the performance of our models using a time-delay strategy was infeasible, so instead we used cross-validation and precision-recall curves. We will have to wait until

the final results are due to be published in October to understand how well our models perform. We do not know whether performance is limited by the RF, our training strategy, or inherent inaccuracies and noise in the features and GO term annotations. Either way, CAFA 4 is likely to generate a large number of high-quality predicted functions for fission yeast, which, if made public, could be hosted on PomBase. However, these predictions will need to be validated by the community.

Preliminary results from CAFA 4 suggest that FunFam-based predictors are still cutting edge, especially amongst tough competition from advanced neural network-based predictors. We were delighted to be placed top for molcular function, as we believe FunFams capture molecular function information well. We were also encouraged by achieving third place for biological process, as these terms are harder to predict using the type of information encoded by FunFams. For comparison, in CAFA 3, we were second for molecular function and fourth for biological process.

5.6.5 Conclusion

Here, we trained machine learning models to predict functions of *S. pombe* proteins. We obtained encouraging results from evaluating our models using cross-validation and also entered our predictions into CAFA 4. However, to be confident about the quality of our predictions, we need experimental validation by growing gene deletion mutant strains in conditions that would elicit loss of function phenotypes. We applied our protein function prediction method to fission yeast as a proof of principle, but the method is species-agnostic, providing feature and target data are available for any species of interest. Despite this, our method is time-consuming to train and is restricted to the information from one species at a time. Going forward, methods that are not restricted to a single species may be more preferable, such as deepFRI [503], which aggregates information from any species.

Aetiologies of protein function at the residue-, domain-, molecular-, cellular-, or organism-level remain a partial mystery, but recent developments in the field of protein function have gone some way towards being able to predict functions. We are grateful to have been able to make a small contribution to the field and its development. In the future, a greater emphasis will be placed on de-blackboxification of predictions and on uncovering general principles that explain how proteins are bestowed with functions.

Chapter 6

Conclusions and future directions

The overarching theme of this thesis was the development and application of protein function prediction methods.

- In Chapter 2, we identified proteins whose expression is significantly altered in an Alzheimer's disease model, focussing on the functional consequences of proteome dysregulation.
- In Chapter 3, we identified putative plastic hydrolase enzymes in metagenomes.
- In Chapter 4, we developed a feature learning method that generates embeddings of proteins according to their multi-network context.
- In Chapter 5, we predicted fission yeast protein functions using network embeddings, evolutionary information from CATH-FunFams and fitness data from phenomic screens of gene deletion mutants.

In this chapter, we identify commonalities between these research projects, draw general conclusions from them, and sketch out future directions of research in these areas.

6.1 Protein function prediction methods that are not restricted to a single species

The work in this thesis made liberal use of high-quality protein network data for protein function prediction (Chapters 2, 4 and 5). These studies focussed on two model organisms—*Schizosaccharomyces pombe* and *Drosophila melanogaster*—that have been the subject of intense research for many decades. Due to the aggregation of information across a large number of orthogonal experiments, we can have a reasonable degree of confidence in network data from such model organisms.

Whilst network-based methods predict protein function well, they do have their

drawbacks. Firstly, they are restricted to organisms that have network data—let alone high-quality data from well characterised species—and are often constrained to be applied to a single organism. Secondly, network-based methods are not applicable to novel data, such as the metagenomic protein sequences we encountered in Chapter 3. Finally, network data can be noisy, as many databases infer edges between proteins from correlations in gene expression, such as from RNA-Seq experiments. Physically-interacting proteins in humans, mice and budding yeast only have a slightly higher correlation in their gene expression than randomly selected pairs of proteins [504]. Despite this, network data has been improving and will continue to do so as high-throughput experiments become more reliable and interactions are confirmed by independent studies.

Ideally, protein function prediction methods would be species-agonostic, such that they are able to use protein information from many different species. One desirable goal is to use *all* protein sequence information. CATH approximates this goal by learning patterns directly from protein structures and sequences, disregarding any associated metadata. The resulting protein family HMMs can be applied to any arbitrary protein sequence to assign the sequence to a family, followed by any functions associated with the family's sequences. We used this method successfully in Chapter 5 to predict GO term annotations in CAFA 4, preliminarily achieving first place for molecular function terms. Sequence embedding methods also achieve this goal by embedding arbitrary length sequences in a fixed dimensional space (Chapter 1). Off-the-shelf supervised machine learning models can be trained on these embeddings to perform function prediction. Such models can be applied to large, diverse protein sequence data sets because all of the sequence information can be integrated via the sequence embedding.

6.2 Determine which types of data and models are most predictive of protein function

Protein function prediction performance is limited by the data used to train predictive models. As the quality, volume and coverage of training data is increased, one would expect model performance to increase. At some point, this relationship may break down as higher-order effects, that are not present in the training data, cannot be accounted for. It is reasonable to assume, however, that we have not reached this point yet. Therefore, improving the training data should in turn improve model performance.

The question then becomes: *which types of data should we use to predict protein function?* It will be extremely useful to understand which data are most predictive of protein function, separately or in combination, to guide experimental and curational data collection efforts going forward. To some extent, the answer depends on what the question is. On one hand, sequence data is ubiquitously available, so can be used to build general function prediction methods (Chapters 3 and 5). Network data, on the other hand, is powerful, but is essentially limited to model organisms, as network data is nonexistant for novel and neglected species (Chapters 2, 4 and 5). Targeted molecular biology and high-throughput screens are possible for culturable species (Chapter 3), but limited to smaller organisms with short life spans (Chapter 5).

In addition, it will be useful to understand which models, given the optimal training data, are most predictive of protein function. Neural networks have shown great promise in recent years and may prove to be the model of choice for protein function prediction. However, whilst neural networks are flexible models, their application to biological sequences is not yet as flexible as HMMs, which have performed well in previous CAFA challenges. Analysis of the best performing methods in the CAFA challenges will help to shed some light on which types of data and which models are most predictive of function. This will be especially true for CAFA 4, for which models would have had a great deal more training data available than for CAFA 3 and neural networks were a more popular choice of method.

6.3 Predicted functions need experimental validation

Predicting functions for proteins is easy; the challenge lies in predicting the correct functions. All predictive methods make trade offs, but on the whole methods wish to minimise the false positive and false negative rates (incorrect predictions) and maximise the true positive and true negative rates (correct predictions). Predictions must be validated by experimental observations that confirm whether the protein performs the predicted function. Experimental validation is useful for confirming true positives and refuting false positives, as these predictions will be contained in the set of predictions generated by a model. However, this strategy is not so useful at identifying true, and false, negatives because these instances may not be contained in the set of predictions. Furthermore, experimental validation can only be applied to functional labels that are actually predicted. There are on the order of $44{,}000$ GO terms, so models are usually trained to predict a subset of these

terms. If a term is not predicted, it cannot be validated.

The work presented in this thesis predicted protein function under protein-centric (Chapters 2, 4 and 5) and function-centric (Chapters 3 and 5) models. These predictions will be validated by our experimental collaborators prior to publication of the work. Our predictions can also be used to guide targeted functional experiments in higher (model) organisms. For example, the proteins that we identified that are dysregulated in Alzheimer's disease in fly brains could be used to design experiments in mice.

Our predictions will help our collaborators to design the functional experiments and phenotypic screens that will be used to validate our predictions. Human intuition and experience will ultimately guide the experiments, according to availablity of resources and the ease with which particular functions can be validated. Jürg Bähler's group at UCL are in the process of validating a selection of our highest confidence fission yeast predictions. Once we have validated these predictions, we will submit our study for publication. In due course, once our group has developed our predictive pipeline for plastic hydrolase sequences, Florian Hollfelder's group at Cambridge will validate these sequences for their efficacy and efficiency in breaking down plastics.

6.4 Expansion of FunFams via new methods and data

In Chapter 3, we introduced FRAN, an algorithmic framework to generate FunFams on arbitrarily large numbers of sequences. This method will be crucial to capture information from the large volumes of diverse proteins that are being sequenced in conventional sequencing projects and metagenomics. Doing so will increase the quality of FunFams because the depth and diversity of FunFam alignments will increase. Better FunFams beget better function predictions, due to increased ability to identify specificity-determining positions that determine particular functions. The need for high-quality and high-coverage protein function predictions continues to grow. With reference to this thesis, better predicted functions could be applied to the yeast proteome (Chapter 5), to uncover other plastic hydrolases (Chapter 3), or to generate more accurate predictions for future CAFA competitions (Chapter 5).

Metagenomics generates unprecedented numbers of protein sequences, many of which are from novel species that live in underrepresented biomes. We would like to capture this information in CATH, Gene3D and FunFams. However, our current methods cannot scale to such behemothic data sets. In the future, our group will use FRAN to gener-

ate FunFams using Gene3D hits from UniProt and MGnify. Doing so will help to improve FunFams and protein functions predicted using them. This is an exciting new direction for CATH, which will help the database and methods to remain competitive when faced with an onslaught of competition from neural network-based methods.

We will use the findings from the analyses performed in the plastic hydrolase project to improve FunFHMMer. The FunFHMMer algorithm was developed, tuned and benchmarked using only three of the superfamiles in CATH [54]. FunFams were then generated for all superfamilies in CATH. Whilst FunFHMMer works well on the three superfamiles used to develop FunFHMMer, and produces high-quality FunFams for these superfamilies, we know that FunFHMMer does not generate such high-quality FunFams for other superfamilies. For example, during our search for novel plastic hydrolases, our analysis of the α/β hydrolase superfamily FunFams has demonstrated that FunFHMMer may be oversplitting sequences into too many FunFams. Compared with CATH v4.2, the latest version, v4.3, has many more FunFams for the α/β hydrolase superfamily. One reason for this may be that v4.3 contains more sequences that are more diverse, so, in turn, these sequences will segregate into more families, each with different SDPs and, therefore, functions. However, we have recently noticed that FunFam alignments tend to have low sequence diversity, as measured by the Neff score for the number of effective sequences in an alignment [18, 505]. In general, sequence diversity in alignments is good for structure prediction, but not for function prediction. As we have recently begun a collaboration that uses FunFams for structure prediction, we are exploring ways to merge FunFams to create 'StructFams' of more diverse sequences that are better for structure prediction. We hope that these improvements will produce better FunFams for all superfamilies.

6.5 Broaden the search for plastic hydrolases

In Chapter 3, we identified putative PET hydrolases in metagenomes. 6% of microbiome samples in MGnify have, so far, been assembled—just the tip of the iceberg—leaving a mountain of information to be mined. The next stage of this project will be to perform targeted assembly of samples from particular biomes to generate more metagenomic protein sequences (Fig. 3.9). We will choose samples from biomes that look promising for finding plastic-degrading enzymes, whether that be from biomes that:

- contain more ABH domains than expected by chance,
- contain proteins with high sequence identity to PETase, or

- from manual examination of biomes by curators.

In Fig. 3.9, we only plotted high-level biomes that do not convey very specific information about the environments that samples were collected from (Section 3.2.1.1). But when selecting samples to be assembled, we will consider more specific biome assignments using the GOLD biome ontology (Section 3.2.1.1) [384]. We will predict proteins from the assembled contigs, which will be analysed for their similarity to PETase and their plastic-degrading potential.

We hope that this iterative pipeline will produce a wealth of information that can be analysed to discover new plastic hydrolases in nature. Putative sequences will be functionally validated using picodroplet functional metagenomics [506] in a collaboration with Florian Hollfelder at Cambridge. We will be able to validate between 10 to 100 sequences using this method. Alongside true positives for positive controls, we will introduce mutations into PETase at key sites, identified using CATH and structural bioinformatics. These mutations may change the efficiency of PET degradation, or even change the function or substrate-specificity to another plastic. Furthermore, we will test putative sequences that we discover through analyses similar to those performed in this work. Our metagenomic search pipeline is not limited to PETases, but is a flexible method to search for proteins that carry out any arbitrary function. The only real requirement is that the function must be able to be validated experimentally to confirm whether sequences do have the predicted functions. We may also explore other enzymatic functions with the Hollfelder group.

References

[1] Bork, P., Dandekar, T., Diaz-Lazcoz, Y., Eisenhaber, F., Huynen, M., and Yuan, Y. "Predicting function: From genes to genomes and back". *Journal of Molecular Biology* (1998).

[2] Friedberg, I. "Automated protein function prediction - The genomic challenge". *Briefings in Bioinformatics* 7.3 (2006), pp. 225–242.

[3] Watson, J. D., Laskowski, R. A., and Thornton, J. M. *Predicting protein function from sequence and structural data.* 2005.

[4] Itakura, K., Hirose, T., Crea, R., Riggs, A. D., Heyneker, H. L., Bolivar, F., and Boyer, H. W. "Expression in Escherichia coli of a chemically synthesized gene for the hormone somatostatin". *Science* (1977).

[5] Apic, G., Gough, J., and Teichmann, S. A. "Domain combinations in archaeal, eubacterial and eukaryotic proteomes". *Journal of Molecular Biology* (2001).

[6] Björklund, Å. K., Ekman, D., Light, S., Frey-Skött, J., and Elofsson, A. "Domain rearrangements in protein evolution". *Journal of Molecular Biology* (2005).

[7] Tian, W. and Skolnick, J. "How well is enzyme function conserved as a function of pairwise sequence identity?" *Journal of Molecular Biology* (2003).

[8] Rost, B. "Enzyme function less conserved than anticipated". *Journal of Molecular Biology* (2002).

[9] Sander, C. and Schneider, R. "Database of homology-derived protein structures and the structural meaning of sequence alignment". *Proteins: Structure, Function, and Bioinformatics* (1991).

[10] Fitch, W. M. "Homology a personal view". *Trends in Genetics* (2000).

[11] Lee, D., Redfern, O., and Orengo, C. "Predicting protein function from sequence and structure". *Nature Reviews Molecular Cell Biology* 8.12 (2007), pp. 995–1005.

[12] Nehrt, N. L., Clark, W. T., Radivojac, P., and Hahn, M. W. "Testing the ortholog conjecture with comparative functional genomic data from mammals". *PLoS Computational Biology* (2011).

[13] Loewenstein, Y., Raimondo, D., Redfern, O. C., Watson, J., Frishman, D., Linial, M., Orengo, C., Thornton, J., and Tramontano, A. "Protein function annotation by homology-based inference." *Genome biology* 10.2 (2009), p. 207.

[14] Altenhoff, A. M., Studer, R. A., Robinson-Rechavi, M., and Dessimoz, C. "Resolving the ortholog conjecture: Orthologs tend to be weakly, but significantly, more similar in function than paralogs". *PLoS Computational Biology* (2012).

[15] Studer, R. A. and Robinson-Rechavi, M. "How confident can we be that orthologs are similar, but paralogs differ?" *Trends in Genetics* (2009).

[16] Stamboulian, M., Guerrero, R. F., Hahn, M. W., and Radivojac, P. "The ortholog conjecture revisited: the value of orthologs and paralogs in function prediction". *Bioinformatics (Oxford, England)* (2020).

[17] Altschul, S. F., Gish, W., Miller, W., Myers, E. W., and Lipman, D. J. "Basic local alignment search tool." *Journal of molecular biology* 215.3 (1990), pp. 403–10.

[18] Altschul, S. F., Madden, T. L., Schäffer, A. A., Zhang, J., Zhang, Z., Miller, W., and Lipman, D. J. "Gapped BLAST and PSI-BLAST: a new generation of protein database search programs." *Nucleic acids research* 25.17 (1997), pp. 3389–402.

[19] Müller, A., MacCallum, R. M., and Sternberg, M. J. "Benchmarking PSI-BLAST in genome annotation". *Journal of Molecular Biology* (1999).

[20] Yu, L., Tanwar, D. K., Penha, E. D. S., Wolf, Y. I., Koonin, E. V., and Basu, M. K. "Grammar of protein domain architectures." *Proceedings of the National Academy of Sciences of the United States of America* 116.9 (2019), pp. 3636–3645.

[21] Lees, J. G., Lee, D., Studer, R. A., Dawson, N. L., Sillitoe, I., Das, S., Yeats, C., Dessailly, B. H., Rentzsch, R., and Orengo, C. A. "Gene3D: Multi-domain annotations for protein sequence and comparative genome analysis". *Nucleic Acids Research* 42.D1 (2014), pp. D240–D245.

[22] Bashton, M. and Chothia, C. "The Generation of New Protein Functions by the Combination of Domains". *Structure* (2007).

[23] Dessailly, B. H., Redfern, O. C., Cuff, A., and Orengo, C. A. "Exploiting structural classifications for function prediction: towards a domain grammar for protein function". *Current Opinion in Structural Biology* 19.3 (2009), pp. 349–356.

[24] El-Gebali, S., Mistry, J., Bateman, A., Eddy, S. R., Luciani, A., Potter, S. C., Qureshi, M., Richardson, L. J., Salazar, G. A., Smart, A., Sonnhammer, E. L., Hirsh, L., Paladin, L., Piovesan, D., Tosatto, S. C., and Finn, R. D. "The Pfam protein families database in 2019". *Nucleic Acids Research* (2019).

[25] Grewal, J. K., Krzywinski, M., and Altman, N. "Markov models — hidden Markov models". *Nature Methods* (2019).

[26] Eddy, S. R. "What is a hidden Markov model?" *Nature Biotechnology* 22.10 (2004), pp. 1315–1316.

[27] Mistry, J., Finn, R. D., Eddy, S. R., Bateman, A., and Punta, M. "Challenges in homology search: HMMER3 and convergent evolution of coiled-coil regions". *Nucleic Acids Research* 41.12 (2013), e121–e121.

[28] Steinegger, M., Meier, M., Mirdita, M., Vöhringer, H., Haunsberger, S. J., and Söding, J. "HH-suite3 for fast remote homology detection and deep protein annotation". *BMC Bioinformatics* (2019).

[29] Söding, J. "Protein homology detection by HMM-HMM comparison". *Bioinformatics* (2005).

[30] Levinthal, C. "How to Fold Graciously". *University of Illinois Press* (1969).

[31] Mizuguchi, K. and Go, N. "Comparison of spatial arrangements of secondary structural elements in proteins". *Protein Engineering* (1995).

[32] Chothia, C. and Lesk, A. "The relation between the divergence of sequence and structure in proteins." *The EMBO Journal* (1986).

[33] Sillitoe, I., Dawson, N., Lewis, T. E., Das, S., Lees, J. G., Ashford, P., Tolulope, A., Scholes, H. M., Senatorov, I., Bujan, A., Ceballos Rodriguez-Conde, F., Dowling, B., Thornton, J., and Orengo, C. A. "CATH: Expanding the horizons of structure-based functional annotations for genome sequences". *Nucleic Acids Research* (2019).

[34] Andreeva, A., Kulesha, E., Gough, J., and Murzin, A. G. "The SCOP database in 2020: Expanded classification of representative family and superfamily domains of known protein structures". *Nucleic Acids Research* (2020).

[35] Grabowski, M., Joachimiak, A., Otwinowski, Z., and Minor, W. "Structural genomics: keeping up with expanding knowledge of the protein universe". *Current Opinion in Structural Biology* (2007).

[36] Glasner, M. E., Gerlt, J. A., and Babbitt, P. C. "Evolution of enzyme superfamilies". *Current Opinion in Chemical Biology* (2006).

[37] Hannenhalli, S. S. and Russell, R. B. "Analysis and prediction of functional subtypes from protein sequence alignments". *Journal of Molecular Biology* (2000).

[38] Orengo, C. A., Pearl, F. M., Bray, J. E., Todd, A. E., Martin, A. C., Lo Conte, L., and Thornton, J. M. "The CATH database provides insights into protein structure/function relationships". *Nucleic Acids Research* (1999).

[39] Orengo, C. A., Michie, A. D., Jones, S., Jones, D. T., Swindells, M. B., and Thornton, J. M. "CATH - A hierarchic classification of protein domain structures". *Structure* (1997).

[40] Taylor, W. R. and Orengo, C. A. "Protein structure alignment". *Journal of Molecular Biology* (1989).

[41] Needleman, S. B. and Wunsch, C. D. "A general method applicable to the search for similarities in the amino acid sequence of two proteins". *Journal of Molecular Biology* 48.3 (1970), pp. 443–453.

[42] Orengo, C. A. and Taylor, W. R. "A local alignment method for protein structure motifs". *Journal of Molecular Biology* (1993).

[43] Smith, T. F. and Waterman, M. S. "Identification of common molecular subsequences". *Journal of Molecular Biology* (1981).

[44] Orengo, C. A. and Taylor, W. R. "SSAP: Sequential structure alignment program for protein structure comparison". *Methods in enzymology*. Vol. 266. 1996, pp. 617–635.

[45] Taylor, W. R. and Orengo, C. A. "A holistic approach to protein structure alignment". *Protein Engineering, Design and Selection* (1989).

[46] Orengo, C. A., Flores, T. P., Taylor, W. R., and Thornton, J. M. "Identification and classification of protein fold families". *Protein Engineering, Design and Selection* (1993).

[47] Redfern, O. C., Harrison, A., Dallman, T., Pearl, F. M., and Orengo, C. A. "CATHEDRAL: A fast and effective algorithm to predict folds and domain boundaries from multidomain protein structures". *PLoS Computational Biology* (2007).

[48] Lees, J., Yeats, C., Perkins, J., Sillitoe, I., Rentzsch, R., Dessailly, B. H., and Orengo, C. "Gene3D: A domain-based resource for comparative genomics, functional annotation and protein network analysis". *Nucleic Acids Research* 40.D1 (2012), pp. D465–D471.

[49] Lewis, T. E., Sillitoe, I., Dawson, N., Lam, S. D., Clarke, T., Lee, D., Orengo, C., and Lees, J. "Gene3D: Extensive prediction of globular domains in proteins". *Nucleic Acids Research* 46.D1 (2018), pp. D435–D439.

[50] Pandurangan, A. P., Stahlhacke, J., Oates, M. E., Smithers, B., and Gough, J. "The SUPERFAMILY 2.0 database: A significant proteome update and a new webserver". *Nucleic Acids Research* (2019).

[51] Fu, L., Niu, B., Zhu, Z., Wu, S., and Li, W. "CD-HIT: Accelerated for clustering the next-generation sequencing data". *Bioinformatics* (2012).

[52] Lewis, T. E., Sillitoe, I., and Lees, J. G. "Cath-resolve-hits: A new tool that resolves domain matches suspiciously quickly". *Bioinformatics* (2019).

[53] Lee, D. A., Rentzsch, R., and Orengo, C. "GeMMA: functional subfamily classification within superfamilies of predicted protein structural domains". *Nucleic Acids Research* 38.3 (2010), pp. 720–737.

[54] Das, S., Lee, D., Sillitoe, I., Dawson, N. L., Lees, J. G., and Orengo, C. A. "Functional classification of CATH superfamilies: a domain-based approach for protein function annotation." *Bioinformaticsc* 31.21 (2015), pp. 3460–7.

[55] Valdar, W. S. "Scoring residue conservation". *Proteins: Structure, Function and Genetics* (2002).

[56] Capra, J. A. and Singh, M. "Characterization and prediction of residues determining protein functional specificity". *Bioinformatics* (2008).

[57] Marks, D. S., Colwell, L. J., Sheridan, R., Hopf, T. A., Pagnani, A., Zecchina, R., and Sander, C. "Protein 3D structure computed from evolutionary sequence variation". *PLoS ONE* (2011).

[58] Senior, A. W., Evans, R., Jumper, J., Kirkpatrick, J., Sifre, L., Green, T., Qin, C., Žídek, A., Nelson, A. W., Bridgland, A., Penedones, H., Petersen, S., Simonyan, K., Crossan, S., Kohli, P., Jones, D. T., Silver, D., Kavukcuoglu, K., and Hassabis, D. "Improved protein structure prediction using potentials from deep learning". *Nature* (2020).

[59] Bairoch, A. "The ENZYME database in 2000". *Nucleic Acids Research* (2000).

[60] Carbon, S., Douglass, E., Dunn, N., Good, B., Harris, N. L., Lewis, S. E., Mungall, C. J., Basu, S., Chisholm, R. L., Dodson, R. J., Hartline, E., Fey, P., Thomas, P. D., Albou, L. P., Ebert, D., Kesling, M. J., Mi, H., Muruganujan, A., Huang, X., Poudel, S., et al. "The Gene Ontology Resource: 20 years and still GOing strong". *Nucleic Acids Research* (2018).

[61] Pagel, P., Kovac, S., Oesterheld, M., Brauner, B., Dunger-Kaltenbach, I., Frishman, G., Montrone, C., Mark, P., Stumpflen, V., Mewes, H.-W., Ruepp, A., and Frishman, D. "The MIPS mammalian protein-protein interaction database". *Bioinformatics* 21.6 (2005), pp. 832–834.

[62] Ruepp, A., Zollner, A., Maier, D., Albermann, K., Hani, J., Mokrejs, M., Tetko, I., Güldener, U., Mannhaupt, G., Münsterkötter, M., and Mewes, H. W. "The FunCat, a functional annotation scheme for systematic classification of proteins from whole genomes". *Nucleic Acids Research* (2004).

[63] Köhler, S., Carmody, L., Vasilevsky, N., Jacobsen, J. O., Danis, D., Gourdine, J. P., Gargano, M., Harris, N. L., Matentzoglu, N., McMurry, J. A., Osumi-Sutherland, D., Cipriani, V., Balhoff, J. P., Conlin, T., Blau, H., Baynam, G., Palmer, R., Gratian, D., Dawkins, H., Segal, M., et al. "Expansion of the Human Phenotype Ontology (HPO) knowledge base and resources". *Nucleic Acids Research* (2019).

[64] Hatos, A., Hajdu-Soltész, B., Monzon, A. M., Palopoli, N., Álvarez, L., Aykac-Fas, B., Bassot, C., Benítez, G. I., Bevilacqua, M., Chasapi, A., Chemes, L., Davey, N. E., Davidović, R., Dunker, A. K., Elofsson, A., Gobeill, J., Foutel, N. S., Sudha, G., Guharoy, M., Horvath, T., et al. "DisProt: Intrinsic protein disorder annotation in 2020". *Nucleic Acids Research* (2020).

[65] Cho, H., Berger, B., and Peng, J. "Compact Integration of Multi-Network Topology for Functional Analysis of Genes". *Cell Systems* (2016).

[66] Gligorijević, V., Barot, M., and Bonneau, R. "deepNF: deep network fusion for protein function prediction". *Bioinformatics* 34.22 (2018). Ed. by Wren, J., pp. 3873–3881.

[67] Kryshtafovych, A., Schwede, T., Topf, M., Fidelis, K., and Moult, J. "Critical assessment of methods of protein structure prediction (CASP)—Round XIII". *Proteins: Structure, Function, and Bioinformatics* 87.12 (2019), pp. 1011–1020.

[68] Haas, J., Barbato, A., Behringer, D., Studer, G., Roth, S., Bertoni, M., Mostaguir, K., Gumienny, R., and Schwede, T. "Continuous Automated Model EvaluatiOn (CAMEO) complementing the critical assessment of structure prediction in CASP12". *Proteins: Structure, Function and Bioinformatics* (2018).

[69] Janin, J. *Welcome to CAPRI: A Critical Assessment of PRedicted Interactions*. 2002.

[70] Janin, J. "Assessing predictions of protein-protein interaction: The CAPRI experiment". *Protein Science* (2005).

[71] Marbach, D., Costello, J. C., Küffner, R., Vega, N. M., Prill, R. J., Camacho, D. M., Allison, K. R., Aderhold, A., Allison, K. R., Bonneau, R., Camacho, D. M., Chen, Y., Collins, J. J., Cordero, F., Costello, J. C., Crane, M., Dondelinger, F., Drton, M., Esposito, R., Foygel, R., et al. "Wisdom of crowds for robust gene network inference". *Nature Methods* 9.8 (2012), pp. 796–804.

[72] Radivojac, P., Clark, W. T., Oron, T. R., Schnoes, A. M., Wittkop, T., Sokolov, A., Graim, K., Funk, C., Verspoor, K., Ben-Hur, A., Pandey, G., Yunes, J. M., Talwalkar, A. S., Repo, S., Souza, M. L., Piovesan, D., Casadio, R., Wang, Z., Cheng, J., Fang, H., et al. "A large-scale evaluation of computational protein function prediction". *Nature Methods* 10.3 (2013), pp. 221–227.

[73] Jiang, Y., Oron, T. R., Clark, W. T., Bankapur, A. R., D'Andrea, D., Lepore, R., Funk, C. S., Kahanda, I., Verspoor, K. M., Ben-Hur, A., Koo, D. C. E., Penfold-Brown, D., Shasha, D., Youngs, N., Bonneau, R., Lin, A., Sahraeian, S. M. E., Martelli, P. L., Profiti, G., Casadio, R., et al. "An expanded evaluation of protein function prediction methods shows an improvement in accuracy". *Genome Biology* 17.1 (2016), p. 184.

[74] Zhou, N., Jiang, Y., Bergquist, T. R., Lee, A. J., Kacsoh, B. Z., Crocker, A. W., Lewis, K. A., Georghiou, G., Nguyen, H. N., Hamid, M. N., Davis, L., Dogan, T., Atalay, V., Rifaioglu, A. S., Dalklran, A., Cetin Atalay, R., Zhang, C., Hurto, R. L., Freddolino, P. L., Zhang, Y., et al. "The CAFA challenge reports improved protein function prediction and new functional annotations for hundreds of genes through experimental screens". *Genome Biology* 20.1 (2019), p. 244.

[75] Hamp, T., Kassner, R., Seemayer, S., Vicedo, E., Schaefer, C., Achten, D., Auer, F., Boehm, A., Braun, T., Hecht, M., Heron, M., Hönigschmid, P., Hopf, T. A., Kaufmann, S., Kiening, M., Krompass, D., Landerer, C., Mahlich, Y., Roos, M., and Rost, B. "Homology-based inference sets the bar high for protein function prediction". *BMC Bioinformatics* (2013).

[76] McCulloch, W. S. and Pitts, W. "A logical calculus of the ideas immanent in nervous activity". *The Bulletin of Mathematical Biophysics* 5.4 (1943), pp. 115–133.

[77] Schmidhuber, J. "Deep learning in neural networks: An overview". *Neural Networks* 61 (2015), pp. 85–117.

[78] Goodfellow, I., Bengio, Y., and Courville, A. *Deep Learning*. MIT Press, 2016.

[79] Lecun, Y., Bengio, Y., and Hinton, G. "Deep learning". *Nature* 521.7553 (2015), pp. 436–444.

[80] Angermueller, C., Pärnamaa, T., Parts, L., and Stegle, O. "Deep learning for computational biology". *Molecular Systems Biology* 12.7 (2016), p. 878.

[81] Min, S., Lee, B., and Yoon, S. "Deep learning in bioinformatics". *Briefings in Bioinformatics* March (2016), bbw068.

[82] Baldi, P. "Deep Learning in Biomedical Data Science". *Annual Review of Biomedical Data Science* 1.1 (2018), pp. 181–205.

[83] Wainberg, M., Merico, D., Delong, A., and Frey, B. J. "Deep learning in biomedicine". *Nature Biotechnology* (2018).

[84] Ching, T., Himmelstein, D. S., Beaulieu-Jones, B. K., Kalinin, A. A., Do, B. T., Way, G. P., Ferrero, E., Agapow, P. M., Zietz, M., Hoffman, M. M., Xie, W., Rosen, G. L., Lengerich, B. J., Israeli, J., Lanchantin, J., Woloszynek, S., Carpenter, A. E., Shrikumar, A., Xu, J., Cofer, E. M., et al. "Opportunities and obstacles for deep learning in biology and medicine". *Journal of the Royal Society Interface* 15.141 (2018).

[85] Bostrom, N. *Superintelligence*. 2017.

[86] Tegmark, M. *Life 3.0*. 2017.

[87] Russel, S. *Human Compatible*. 2019.

[88] Lin, H. W., Tegmark, M., and Rolnick, D. "Why Does Deep and Cheap Learning Work So Well?" *Journal of Statistical Physics* 168.6 (2017), pp. 1223–1247.

[89] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. "Learning representations by back-propagating errors". *Nature* 323.6088 (1986), pp. 533–536.

[90] Kingma, D. P. and Ba, J. "Adam: A Method for Stochastic Optimization" (2014).

[91] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". 15 (2014), pp. 1929–1958.

[92] Hamilton, W. L., Ying, R., and Leskovec, J. "Representation Learning on Graphs: Methods and Applications" (2017).

[93] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D. "Backpropagation Applied to Handwritten Zip Code Recognition". *Neural Computation* 1.4 (1989), pp. 541–551.

[94] Krizhevsky, A. and Hinton, G. E. "ImageNet Classification with Deep Convolutional Neural Networks". *NIPS'12 Proceedings of the 25th International Conference* 1 (2012), pp. 1–9.

[95] Bai, S., Kolter, J. Z., and Koltun, V. "An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling" (2018).

[96] Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R., and Bengio, Y. "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention" (2015).

[97] Elbayad, M., Besacier, L., and Verbeek, J. "Pervasive Attention: 2D Convolutional Neural Networks for Sequence-to-Sequence Prediction" (2018).

[98] Bengio, Y., Simard, P., and Frasconi, P. "Learning long-term dependencies with gradient descent is difficult". *IEEE Transactions on Neural Networks* 5.2 (1994), pp. 157–166.

[99] Hochreiter, S. and Schmidhuber, J. "Long Short-Term Memory". *Neural Computation* 9.8 (1997), pp. 1735–1780.

[100] Hyafil, L. and Rivest, R. "Constructing Optimal Binary Search Trees is NP Complete". *Information Processing Letters* (1976).

[101] Probst, P., Wright, M., and Boulesteix, A.-L. "Hyperparameters and Tuning Strategies for Random Forest" (2018).

[102] Cristianini, N. and Shawe-Taylor, J. *Kernel Methods for Pattern Analysis*. 2004.

[103] Cai, H., Zheng, V. W., and Chang, K. C. C. "A Comprehensive Survey of Graph Embedding: Problems, Techniques, and Applications". *IEEE Transactions on Knowledge and Data Engineering* 30.9 (2018), pp. 1616–1637.

[104] Grover, A. and Leskovec, J. "node2vec: Scalable Feature Learning for Networks" (2016).

[105] Perozzi, B., Al-Rfou, R., and Skiena, S. "DeepWalk: Online Learning of Social Representations" (2014), pp. 701–710.

[106] Cao, S., Lu, W., and Xu, Q. "Deep neural networks for learning graph representations". *30th AAAI Conference on Artificial Intelligence, AAAI 2016*. 2016.

[107] Wang, D., Cui, P., and Zhu, W. "Structural deep network embedding". *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2016.

[108] Franz, M., Rodriguez, H., Lopes, C., Zuberi, K., Montojo, J., Bader, G. D., and Morris, Q. "GeneMANIA update 2018". *Nucleic Acids Research* (2018).

[109] Dutkowski, J., Kramer, M., Surma, M. a., Balakrishnan, R., Cherry, J. M., Krogan, N. J., and Ideker, T. "A gene ontology inferred from molecular networks." *Nature biotechnology* 31.1 (2013), pp. 38–45.

[110] Yu, M. K., Kramer, M., Dutkowski, J., Srivas, R., Licon, K., Kreisberg, J. F., Ng, C. T., Krogan, N., Sharan, R., and Ideker, T. "Translation of genotype to phenotype by a hierarchy of cell subsystems". *Cell Systems* (2016).

[111] Jerby-Arnon, L., Pfetzer, N., Waldman, Y. Y., McGarry, L., James, D., Shanks, E., Seashore-Ludlow, B., Weinstock, A., Geiger, T., Clemons, P. A., Gottlieb, E., and Ruppin, E. "Predicting cancer-specific vulnerability via data-driven detection of synthetic lethality". *Cell* (2014).

[112] Zitnik, M. and Leskovec, J. "Predicting multicellular function through multi-layer tissue networks". *Bioinformatics*. 2017.

[113] Tang, J., Qu, M., Wang, M., Zhang, M., Yan, J., and Mei, Q. "LINE: Large-scale information network embedding". *WWW 2015 - Proceedings of the 24th International Conference on World Wide Web*. 2015.

[114] Chen, H., Hu, Y., Perozzi, B., and Skiena, S. "HARP: Hierarchical representation learning for networks". *32nd AAAI Conference on Artificial Intelligence, AAAI 2018*. 2018.

[115] Fouss, F., Francoisse, K., Yen, L., Pirotte, A., and Saerens, M. "An experimental investigation of kernels on graphs for collaborative recommendation and semisupervised classification". *Neural Networks* 31 (2012), pp. 53–72.

[116] Heriche, J.-K., Lees, J. G., Morilla, I., Walter, T., Petrova, B., Roberti, M. J., Hossain, M. J., Adler, P., Fernandez, J. M., Krallinger, M., Haering, C. H., Vilo, J., Valencia, A., Ranea, J. A., Orengo, C., and Ellenberg, J. "Integration of biological data by kernels on graph nodes allows prediction of new genes involved in mitotic chromosome condensation". *Molecular Biology of the Cell* 25.16 (2014), pp. 2522–2536.

[117] Lehtinen, S., Lees, J., Bähler, J., Shawe-Taylor, J., and Orengo, C. "Gene function prediction from functional association networks using kernel partial least squares regression". *PLoS ONE* 10.8 (2015), pp. 1–14.

[118] Sigoillot, F. D. and King, R. W. "Vigilance and validation: Keys to success in RNAi screening". *ACS Chemical Biology* 6.1 (2011), pp. 47–60.

[119] Mering, C. von, Jensen, L. J., Snel, B., Hooper, S. D., Krupp, M., Foglierini, M., Jouffre, N., Huynen, M. A., and Bork, P. "STRING: known and predicted protein-protein associations, integrated and transferred across organisms." *Nucleic acids research* 33.Database issue (2005), pp. D433–7.

[120] Mostafavi, S., Ray, D., Warde-Farley, D., Grouios, C., and Morris, Q. "GeneMANIA: A real-time multiple association network integration algorithm for predicting gene function". *Genome Biology* 9.SUPPL. 1 (2008), S4.

[121] Saerens, M., Fouss, F., Yen, L., and Dupont, P. "The principal components analysis of a graph, and its relationships to spectral clustering". *Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science)*. 2004.

[122] Bach, F. R. and Jordan, M. I. "Learning spectral clustering". *Advances in Neural Information Processing Systems*. 2004.

[123] Das, S., Scholes, H. M., Sen, N., and Orengo, C. "CATH functional families predict functional sites in proteins". *Bioinformatics* (2020). Ed. by Elofsson, A., pp. 0–0.

[124] Zhang, D. and Kabuka, M. "Protein Family Classification from Scratch: A CNN based Deep Learning Approach". *IEEE/ACM Transactions on Computational Biology and Bioinformatics* (2020).

[125] Seo, S., Oh, M., Park, Y., and Kim, S. "DeepFam: deep learning based alignment-free method for protein family modeling and prediction". *Bioinformatics* 34.13 (2018), pp. i254–i262.

[126] Alipanahi, B., Delong, A., Weirauch, M. T., and Frey, B. J. "Predicting the sequence specificities of DNA- and RNA-binding proteins by deep learning". *Nature Biotechnology* 33.8 (2015), pp. 831–838.

[127] Kelley, D. R., Snoek, J., and Rinn, J. L. "Basset: learning the regulatory code of the accessible genome with deep convolutional neural networks." *Genome research* 26.7 (2016), pp. 990–9.

[128] Quang, D. and Xie, X. "DanQ: a hybrid convolutional and recurrent deep neural network for quantifying the function of DNA sequences". *Nucleic Acids Research* 44.11 (2016), e107–e107.

[129] Zhou, J. and Troyanskaya, O. G. "Predicting effects of noncoding variants with deep learning-based sequence model". *Nature Methods* 12.10 (2015), pp. 931–934.

[130] Angermueller, C., Lee, H. J., Reik, W., and Stegle, O. "DeepCpG: Accurate prediction of single-cell DNA methylation states using deep learning". *Genome Biology* (2017).

[131] Mikolov, T., Chen, K., Corrado, G., and Dean, J. *Distributed Representations of Words and Phrases and their Compositionality*. Tech. rep.

[132] Le, Q. V. and Mikolov, T. "Distributed Representations of Sentences and Documents". *31st International Conference on Machine Learning, ICML 2014* 4 (2014), pp. 2931–2939.

[133] Ren, J., Bai, X., Lu, Y. Y., Tang, K., Wang, Y., Reinert, G., and Sun, F. "Alignment-Free Sequence Analysis and Applications". *Annual Review of Biomedical Data Science* 8.1 (2018).

[134] Ondov, B. D., Treangen, T. J., Melsted, P., Mallonee, A. B., Bergman, N. H., Koren, S., and Phillippy, A. M. "Mash: Fast genome and metagenome distance estimation using MinHash". *Genome Biology* 17.1 (2016), p. 132.

[135] Buhler, J. "Efficient large-scale sequence comparison by locality-sensitive hashing". *Bioinformatics* 17.5 (2001), pp. 419–428.

[136] Luo, Y., Yu, Y. W., Zeng, J., Berger, B., and Peng, J. "Metagenomic binning through low-density hashing". *Bioinformatics* (2019).

[137] Steinegger, M. and Söding, J. "Clustering huge protein sequence sets in linear time". *Nature Communications* 9.1 (2018), p. 2542.

[138] Salakhutdinov, R. and Hinton, G. "Semantic hashing". *International Journal of Approximate Reasoning* (2009).

[139] Asgari, E. and Mofrad, M. R. K. "ProtVec: A Continuous Distributed Representation of Biological Sequences". *PLoS ONE* 10.11 (2015).

[140] Ng, P. "dna2vec: Consistent vector representations of variable-length k-mers" (2017).

[141] Cai, Y., Wang, J., and Deng, L. "SDN2GO: An Integrated Deep Learning Model for Protein Function Prediction". *Frontiers in Bioengineering and Biotechnology* 8 (2020), p. 391.

[142] Asgari, E., McHardy, A. C., and Mofrad, M. R. "Probabilistic variable-length segmentation of protein sequences for discriminative motif discovery (DiMotif) and sequence embedding (ProtVecX)". *Scientific Reports* (2019).

[143] Kimothi, D., Soni, A., Biyani, P., and Hogan, J. M. "Distributed Representations for Biological Sequence Analysis" (2016).

[144] Melidis, D. P., Malone, B., and Nejdl, W. "dom2vec: Assessable domain embeddings and their use for protein prediction tasks". *bioRxiv* (2020).

[145] Heinzinger, M., Elnaggar, A., Wang, Y., Dallago, C., Nechaev, D., Matthes, F., and Rost, B. "Modeling aspects of the language of life through transfer-learning protein sequences". *BMC Bioinformatics* (2019).

[146] Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. "Deep contextualized word representations". *NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*. 2018.

[147] Alley, E. C., Khimulya, G., Biswas, S., AlQuraishi, M., and Church, G. M. "Unified rational protein engineering with sequence-based deep representation learning". *Nature Methods* (2019).

[148] Hou, J., Adhikari, B., and Cheng, J. "DeepSF: deep convolutional neural network for mapping protein sequences to folds". *Bioinformatics* 34.8 (2018). Ed. by Valencia, A., pp. 1295–1303.

[149] Bileschi, M. L., Belanger, D., Bryant, D., Sanderson, T., Carter, B., Sculley, D., DePristo, M. A., and Colwell, L. J. "Using Deep Learning to Annotate the Protein Universe". *bioRxiv* (2019), p. 626507.

[150] Zhang, D. and Kabuka, M. "Multimodal deep representation learning for protein interaction identification and protein family classification". *BMC Bioinformatics* (2019).

[151] Jin, C. and Cukier, R. I. "Machine learning can be used to distinguish protein families and generate new proteins belonging to those families". *Journal of Chemical Physics* (2019).

[152] Riesselman, A. J., Ingraham, J. B., and Marks, D. S. "Deep generative models of genetic variation capture the effects of mutations". *Nature Methods* 15.10 (2018), pp. 816–822.

[153] Dong, Y., Chawla, N. V., and Swami, A. "Metapath2vec: Scalable representation learning for heterogeneous networks". *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2017.

[154] Salwinski, L. "The Database of Interacting Proteins: 2004 update". *Nucleic Acids Research* (2004).

[155] Keshava Prasad, T. S., Goel, R., Kandasamy, K., Keerthikumar, S., Kumar, S., Mathivanan, S., Telikicherla, D., Raju, R., Shafreen, B., Venugopal, A., Balakrishnan, L., Marimuthu, A., Banerjee, S., Somanathan, D. S., Sebastian, A., Rani, S., Ray, S., Har-

rys Kishore, C. J., Kanth, S., Ahmed, M., et al. "Human Protein Reference Database - 2009 update". *Nucleic Acids Research* (2009).

[156] You, R., Huang, X., and Zhu, S. "DeepText2GO: Improving large-scale protein function prediction with deep semantic text representation". *Methods* (2018).

[157] Fa, R., Cozzetto, D., Wan, C., and Jones, D. T. "Predicting human protein function with multitask deep neural networks". *PLoS ONE* (2018).

[158] Kulmanov, M., Khan, M. A., and Hoehndorf, R. "DeepGO: Predicting protein functions from sequence and interactions using a deep ontology-aware classifier". *Bioinformatics* (2018).

[159] Alshahrani, M., Khan, M. A., Maddouri, O., Kinjo, A. R., Queralt-Rosinach, N., and Hoehndorf, R. "Neuro-symbolic representation learning on biological knowledge graphs". *Bioinformatics* 33.17 (2017), pp. 2723–2730.

[160] Kulmanov, M., Hoehndorf, R., and Cowen, L. "DeepGOPlus: Improved protein function prediction from sequence". *Bioinformatics* (2020).

[161] Cao, R., Freitas, C., Chan, L., Sun, M., Jiang, H., and Chen, Z. "ProLanGO: Protein function prediction using neural machine translation based on a recurrent neural network". *Molecules* (2017).

[162] Sinai, S., Kelsic, E., Church, G. M., and Nowak, M. A. "Variational auto-encoding of protein sequences" (2018).

[163] Scholes, H. M., Cryar, A., Kerr, F., Sutherland, D., Gethings, L. A., Vissers, J. P. C., Lees, J. G., Orengo, C. A., Partridge, L., and Thalassinos, K. "Dynamic changes in the brain protein interaction network correlates with progression of Aβ42 pathology in Drosophila". *Scientific Reports* 10.1 (2020), p. 18517.

[164] Lane, C. A., Hardy, J., and Schott, J. M. "Alzheimer's disease". *European Journal of Neurology* 25.1 (2018), pp. 59–70.

[165] Alzheimer, A. "Über eine eigenartige Erkrankung der Hirnrinde". *Allgemeine Zeitschrift für Psychiatrie und psychisch-gerichtliche Medizin* 64 (1907), pp. 146–148.

[166] Glenner, G. G. and Wong, C. W. "Alzheimer's disease: initial report of the purification and characterization of a novel cerebrovascular amyloid protein. 1984." *Biochemical and biophysical research communications* 425.3 (2012), pp. 534–539.

[167] Grundke-Iqbal, I., Iqbal, K., Tung, Y. C., Quinlan, M., Wisniewski, H. M., and Binder, L. I. "Abnormal phosphorylation of the microtubule-associated protein tau in Alzheimer cytoskeletal pathology." *Proceedings of the National Academy of Sciences of the United States of America* 83.13 (1986), pp. 4913–7.

[168] Goedert, M., Wischik, C. M., Crowther, R. A., Walker, J. E., and Klug, A. "Cloning and sequencing of the cDNA encoding a core protein of the paired helical filament of Alzheimer disease: identification as the microtubule-associated protein tau." *Proceedings of the National Academy of Sciences of the United States of America* 85.11 (1988), pp. 4051–4055.

[169] Cai, H., Cong, W.-n., Ji, S., Rothman, S., Maudsley, S., and Martin, B. "Metabolic Dysfunction in Alzheimers Disease and Related Neurodegenerative Disorders". *Current Alzheimer Research* 9.1 (2012), pp. 5–17.

[170] Szutowicz, A., Bielarczyk, H., Jankowska-Kulawy, A., Pawełczyk, T., and Ronowska, A. "Acetyl-CoA the key factor for survival or death of cholinergic neurons in course of neurodegenerative diseases." *Neurochemical research* 38.8 (2013), pp. 1523–42.

[171] Suberbielle, E., Sanchez, P. E., Kravitz, A. V., Wang, X., Ho, K., Eilertson, K., Devidze, N., Kreitzer, A. C., and Mucke, L. "Physiologic brain activity causes DNA double-

strand breaks in neurons, with exacerbation by amyloid-β". *Nature Neuroscience* (2013).

[172] Raina, A. K., Monteiro, M. J., Mcshea, A., and Smith, M. A. "The role of cell cycle-mediated events in Alzheimer's disease". *International Journal of Experimental Pathology* 80.2 (1999), pp. 71–76.

[173] Kanaan, N. M., Pigino, G. F., Brady, S. T., Lazarov, O., Binder, L. I., and Morfini, G. A. "Axonal degeneration in Alzheimer's disease: When signaling abnormalities meet the axonal transport system". *Experimental Neurology* 246 (2013), pp. 44–53.

[174] Donev, R., Kolev, M., Millet, B., and Thome, J. "Neuronal death in Alzheimer's disease and therapeutic opportunities". *Journal of Cellular and Molecular Medicine* (2009).

[175] Van Cauwenberghe, C., Van Broeckhoven, C., and Sleegers, K. "The genetic landscape of Alzheimer disease: clinical implications and perspectives". *Genetics in Medicine* 18.5 (2016), pp. 421–430.

[176] Müller, U. C., Deller, T., and Korte, M. "Not just amyloid: Physiological functions of the amyloid precursor protein family". *Nature Reviews Neuroscience* 18.5 (2017), pp. 281–298.

[177] Zhang, Y., McLaughlin, R., Goodyer, C., and LeBlanc, A. "Selective cytotoxicity of intracellular amyloid beta peptide1-42 through p53 and Bax in cultured primary human neurons." *The Journal of cell biology* 156.3 (2002), pp. 519–29.

[178] McGowan, E., Pickford, F., Kim, J., Onstead, L., Eriksen, J., Yu, C., Skipper, L., Murphy, M. P., Beard, J., Das, P., Jansen, K., DeLucia, M., Lin, W. L., Dolios, G., Wang, R., Eckman, C. B., Dickson, D. W., Hutton, M., Hardy, J., and Golde, T. "Aβ42 is essential for parenchymal and vascular amyloid deposition in mice". *Neuron* 47.2 (2005), pp. 191–199.

[179] Götz, J., Chen, F., Dorpe, J. van, and Nitsch, R. M. "Formation of neurofibrillary tangles in P301l tau transgenic mice induced by Aβ42 fibrils." *Science (New York, N.Y.)* 293.5534 (2001), pp. 1491–5.

[180] Uhlén, M., Fagerberg, L., Hallström, B. M., Lindskog, C., Oksvold, P., Mardinoglu, A., Sivertsson, Å., Kampf, C., Sjöstedt, E., Asplund, A., Olsson, I., Edlund, K., Lundberg, E., Navani, S., Szigyarto, C. A.-k., Odeberg, J., Djureinovic, D., Takanen, J. O., Hober, S., Alm, T., et al. "Tissue-based map of the human proteome". *Science (New York, N.Y.)* 347.6220 (2015), pp. 1260419–1260419.

[181] Murayama, Y., Takeda, S., Yonezawa, K., Giambarella, U., Nishimoto, I., and Ogata, E. "Cell surface receptor function of amyloid precursor protein that activates Ser/Thr kinases". *Gerontology* 42.Suppl 1 (1996), pp. 2–11.

[182] Shu, R., Wong, W., Ma, Q. H., Yang, Z. Z., Zhu, H., Liu, F. J., Wang, P., Ma, J., Yan, S., Polo, J. M., Bernard, C. C. A., Stanton, L. W., Dawe, G. S., and Xiao, Z. C. "APP intracellular domain acts as a transcriptional regulator of miR-663 suppressing neuronal differentiation". *Cell Death and Disease* 6.2 (2015), e1651.

[183] Chow, V. W., Mattson, M. P., Wong, P. C., and Gleichmann, M. "An overview of APP processing enzymes and products." *Neuromolecular medicine* 12.1 (2010), pp. 1–12.

[184] LaFerla, F. M., Green, K. N., and Oddo, S. "Intracellular amyloid-β in Alzheimer's disease". *Nature Reviews Neuroscience* 8.7 (2007), pp. 499–509.

[185] Jarrett, J. T., Berger, E. P., and Lansbury, P. T. "The carboxy terminus of the beta amyloid protein is critical for the seeding of amyloid formation: implications for the pathogenesis of Alzheimer's disease." *Biochemistry* 32.18 (1993), pp. 4693–7.

[186] Mullan, M., Crawford, F., Axelman, K., Houlden, H., Lilius, L., Winblad, B., and Lannfelt, L. "A pathogenic mutation for probable Alzheimer's disease in the APP gene at the N–terminus of β–amyloid". *Nature Genetics* 1.5 (1992), pp. 345–347.

[187] Nilsberth, C., Westlind-Danielsson, A., Eckman, C. B., Condron, M. M., Axelman, K., Forsell, C., Stenh, C., Luthman, J., Teplow, D. B., Younkin, S. G., Näslund, J., and Lannfelt, L. "The 'Arctic' APP mutation (E693G) causes Alzheimer's disease by enhanced Aβ protofibril formation". *Nature Neuroscience* 4.9 (2001), pp. 887–893.

[188] Serpell, L. C. "Alzheimer's amyloid fibrils: structure and assembly". *Biochimica et Biophysica Acta (BBA) - Molecular Basis of Disease* 1502.1 (2000), pp. 16–30.

[189] Ahmed, M., Davis, J., Aucoin, D., Sato, T., Ahuja, S., Aimoto, S., Elliott, J. I., Van Nostrand, W. E., and Smith, S. O. "Structural conversion of neurotoxic amyloid-β1–42 oligomers to fibrils". *Nature Structural & Molecular Biology* 17.5 (2010), pp. 561–567.

[190] Miller, D. L., Papayannopoulos, I. A., Styles, J., Bobin, S. A., Lin, Y. Y., Biemann, K., and Iqbal, K. "Peptide compositions of the cerebrovascular and senile plaque core amyloid deposits of Alzheimer's disease." *Archives of biochemistry and biophysics* 301.1 (1993), pp. 41–52.

[191] Drummond, E., Nayak, S., Faustin, A., Pires, G., Hickman, R. A., Askenazi, M., Cohen, M., Haldiman, T., Kim, C., Han, X., Shao, Y., Safar, J. G., Ueberheide, B., and Wisniewski, T. "Proteomic differences in amyloid plaques in rapidly progressive and sporadic Alzheimer's disease". *Acta Neuropathologica* (2017).

[192] Younkin, S. G. "The role of Aβ42 in Alzheimer's disease". *Journal of Physiology Paris* 92.3-4 (1998), pp. 289–292.

[193] Hatami, A., Monjazeb, S., Milton, S., and Glabe, C. G. "Familial Alzheimer's Disease Mutations within the Amyloid Precursor Protein Alter the Aggregation and Conformation of the Amyloid-β Peptide." *The Journal of biological chemistry* 292.8 (2017), pp. 3172–3185.

[194] Murakami, K., Irie, K., Morimoto, A., Ohigashi, H., Shindo, M., Nagao, M., Shimizu, T., and Shirasawa, T. "Synthesis, aggregation, neurotoxicity, and secondary structure of various Aβ1–42 mutants of familial Alzheimer's disease at positions 21–23". *Biochemical and Biophysical Research Communications* 294.1 (2002), pp. 5–10.

[195] Masters, C. L., Multhaup, G., Simms, G., Pottgiesser, J., Martins, R. N., and Beyreuther, K. "Neuronal origin of a cerebral amyloid: neurofibrillary tangles of Alzheimer's disease contain the same protein as the amyloid of plaque cores and blood vessels." *The EMBO journal* 4.11 (1985), pp. 2757–2763.

[196] Gouras G K, Tsai J, Naslund J, Vincent B, Edgar M, Checler F, Greenfield J P, Haroutunian V, Buxbaum J D, Xu H, Greengard P, and Relkin N R. "Intraneuronal Aβ42 Accumulation in Human Brain". *Am. J. Pathol.* 156.1 (2000), pp. 15–20.

[197] Weingarten, M. D., Lockwood, A. H., Hwo, S. Y., and Kirschner, M. W. "A protein factor essential for microtubule assembly." *Proceedings of the National Academy of Sciences of the United States of America* 72.5 (1975), pp. 1858–62.

[198] Iqbal, K., Liu, F., Gong, C.-X., and Grundke-Iqbal, I. "Tau in Alzheimer disease and related tauopathies." *Current Alzheimer research* 7.8 (2010), pp. 656–64.

[199] Alonso, A. d. C., Li, B., Grundke-Iqbal, I., and Iqbal, K. "Polymerization of hyperphosphorylated tau into filaments eliminates its inhibitory activity". *Proceedings of the National Academy of Sciences* 103.23 (2006), pp. 8864–8869.

[200] Alonso, A. C., Zaidi, T., Grundke-Iqbal, I., and Iqbal, K. "Role of abnormally phosphorylated tau in the breakdown of microtubules in Alzheimer disease." *Proceedings of the National Academy of Sciences of the United States of America* 91.12 (1994), pp. 5562–6.

[201] Moya-Alvarado, G., Gershoni-Emek, N., Perlson, E., and Bronfman, F. C. "Neurodegeneration and Alzheimer's disease (AD). What can proteomics tell us about the Alzheimer's brain?" *Molecular and Cellular Proteomics* 15.2 (2016), pp. 409–425.

[202] Lynn, B. C., Wang, J., Markesbery, W. R., and Lovell, M. A. "Quantitative changes in the mitochondrial proteome from subjects with mild cognitive impairment, early stage, and late stage Alzheimer's disease". *Journal of Alzheimer's Disease* 19.1 (2010), pp. 325–339.

[203] Butterfield, D. A., Di Domenico, F., Swomley, A. M., Head, E., and Perluigi, M. "Redox proteomics analysis to decipher the neurobiology of Alzheimer-like neurodegeneration: Overlaps in Down's syndrome and Alzheimer's disease brain". *Biochemical Journal* 463.2 (2014), pp. 177–189.

[204] Aluise, C. D., Robinson, R. A., Cai, J., Pierce, W. M., Markesbery, W. R., and Butterfield, D. A. "Redox proteomics analysis of brains from subjects with amnestic mild cognitive impairment compared to brains from subjects with preclinical alzheimer's disease: Insights into memory loss in MCI". *Journal of Alzheimer's Disease* 23.2 (2011), pp. 257–269.

[205] Dammer, E. B., Lee, A. K., Duong, D. M., Gearing, M., Lah, J. J., Levey, A. I., and Seyfried, N. T. "Quantitative phosphoproteomics of Alzheimer's disease reveals cross-talk between kinases and small heat shock proteins". *Proteomics* 15.2-3 (2015), pp. 508–519.

[206] Sultana, R., Robinson, R. A., Di Domenico, F., Abdul, H. M., St. Clair, D. K., Markesbery, W. R., Cai, J., Pierce, W. M., and Butterfield, D. A. "Proteomic identification of specifically carbonylated brain proteins in APP NLh/APP NLh×PS-1 P264L/PS-1 P264L human double mutant knock-in mice model of Alzheimer disease as a function of age". *Journal of Proteomics* 74.11 (2011), pp. 2430–2440.

[207] Sofola, O., Kerr, F., Rogers, I., Killick, R., Augustin, H., Gandy, C., Allen, M. J., Hardy, J., Lovestone, S., and Partridge, L. "Inhibition of GSK-3 Ameliorates Aβ Pathology in an Adult-Onset Drosophila Model of Alzheimer's Disease". *PLoS Genetics* 6.9 (2010). Ed. by Lu, B., e1001087.

[208] Keifer, D. Z., Motwani, T., Teschke, C. M., and Jarrold, M. F. "Measurement of the accurate mass of a 50 MDa infectious virus". *Rapid communications in mass spectrometry : RCM* 30.17 (2016), pp. 1957–1962.

[209] Wasinger, V. C., Cordwell, S. J., Poljak, A., Yan, J. X., Gooley, A. A., Wilkins, M. R., Duncan, M. W., Harris, R., Williams, K. L., and Humphery-Smith, I. "Progress with gene-product mapping of the Mollicutes: Mycoplasma genitalium". *Electrophoresis* 16.1 (1995), pp. 1090–1094.

[210] Aebersold, R. and Mann, M. *Mass spectrometry-based proteomics.* 2003.

[211] Tanaka, K., Waki, H., Ido, Y., Akita, S., Yoshida, Y., Yoshida, T., and Matsuo, T. "Protein and polymer analyses up to m/z 100 000 by laser ionization time-of-flight mass spectrometry". *Rapid Communications in Mass Spectrometry* 2.8 (1988), pp. 151–153.

[212] Karas, M., Bachmann, D., and Hillenkamp, F. "Influence of the Wavelength in High-Irradiance Ultraviolet Laser Desorption Mass Spectrometry of Organic Molecules". *Analytical Chemistry* 57.14 (1985), pp. 2935–2939.

[213] Whitehouse, C. M., Dreyer, R. N., Yamashita, M., and Fenn, J. B. "Electrospray Interface for Liquid Chromatographs and Mass Spectrometers". *Analytical Chemistry* 57.3 (1985), pp. 675–679.

[214] Gabelica, V. and Marklund, E. "Fundamentals of ion mobility spectrometry". *Current Opinion in Chemical Biology* 42 (2018), pp. 51–59.

[215] May, J. C., Morris, C. B., and McLean, J. A. "Ion mobility collision cross section compendium". *Analytical Chemistry* 89.2 (2017), pp. 1032–1044.

[216] Schubert, O. T., Röst, H. L., Collins, B. C., Rosenberger, G., and Aebersold, R. "Quantitative proteomics: Challenges and opportunities in basic and applied research". *Nature Protocols* 12.7 (2017), pp. 1289–1294.

[217] Noor, Z., Ahn, S. B., Baker, M. S., Ranganathan, S., and Mohamedali, A. "Mass spectrometry–based protein identification in proteomics—a review". *Briefings In Bioinformatics* (2019).

[218] Henderson, R. A., Michel, H., Sakaguchi, K., Shabanowitz, J., Appella, E., Hunt, D. F., and Engelhard, V. H. "HLA-A2.1-Associated peptides from a mutant cell line: A second pathway of antigen presentation". *Science* 255.5049 (1992), pp. 1264–1266.

[219] Gillet, L. C., Leitner, A., and Aebersold, R. "Mass Spectrometry Applied to Bottom-Up Proteomics: Entering the High-Throughput Era for Hypothesis Testing". *Annual Review of Analytical Chemistry* 9.1 (2016), pp. 449–472.

[220] Chapman, J. D., Goodlett, D. R., and Masselon, C. D. "Multiplexed and data-independent tandem mass spectrometry for global proteome profiling". *Mass Spectrometry Reviews* 33.6 (2014), pp. 452–470.

[221] Geromanos, S. J., Hughes, C., Ciavarini, S., Vissers, J. P., and Langridge, J. I. "Using ion purity scores for enhancing quantitative accuracy and precision in complex proteomics samples". *Analytical and Bioanalytical Chemistry* 404.4 (2012), pp. 1127–1139.

[222] Crowther, D. C., Kinghorn, K. J., Miranda, E., Page, R., Curry, J. A., Duthie, F. A. I., Gubb, D. C., and Lomas, D. A. "Intraneuronal Aβ, non-amyloid aggregates and neurodegeneration in a Drosophila model of Alzheimer's disease". *Neuroscience* 132.1 (2005), pp. 123–135.

[223] Osterwalder, T., Yoon, K. S., White, B. H., and Keshishian, H. "A conditional tissue-specific transgene expression system using inducible GAL4." *Proceedings of the National Academy of Sciences of the United States of America* 98.22 (2001), pp. 12596–12601.

[224] Li, G. Z., Vissers, J. P., Silva, J. C., Golick, D., Gorenstein, M. V., and Geromanos, S. J. "Database searching and accounting of multiplexed precursor and product ion spectra from the data independent analysis of simple and complex peptide mixtures". *Proteomics* 9.6 (2009), pp. 1696–1719.

[225] Distler, U., Kuharev, J., Navarro, P., Levin, Y., Schild, H., and Tenzer, S. "Drift time-specific collision energies enable deep-coverage data-independent acquisition proteomics". *Nature Methods* 11.2 (2014), pp. 167–170.

[226] Silva, J. C., Gorenstein, M. V., Li, G.-Z., Vissers, J. P. C., and Geromanos, S. J. "Absolute quantification of proteins by LCMSE: a virtue of parallel MS acquisition." *Molecular & cellular proteomics : MCP* 5.1 (2006), pp. 144–56.

[227] Lazar, C., Gatto, L., Ferro, M., Bruley, C., and Burger, T. "Accounting for the Multiple Natures of Missing Values in Label-Free Quantitative Proteomics Data Sets to Compare Imputation Strategies". *Journal of Proteome Research* 15.4 (2016), pp. 1116–1125.

[228] Bolstad, B. M., Irizarry, R. A., Åstrand, M., and Speed, T. P. "A comparison of normalization methods for high density oligonucleotide array data based on variance and bias". *Bioinformatics* 19.2 (2003), pp. 185–193.

[229] Love, M. I., Huber, W., and Anders, S. "Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2". *Genome Biology* 15.12 (2014), p. 550.

[230] Woo, S., Leek, J. T., and Storey, J. D. "A computationally efficient modular optimal discovery procedure". *Bioinformatics* 27.4 (2011), pp. 509–515.

[231] Robinson, M. D., McCarthy, D. J., and Smyth, G. K. "edgeR: A Bioconductor package for differential expression analysis of digital gene expression data". *Bioinformatics* 26.1 (2009), pp. 139–140.

[232] Ritchie, M. E., Phipson, B., Wu, D., Hu, Y., Law, C. W., Shi, W., and Smyth, G. K. "Limma powers differential expression analyses for RNA-sequencing and microarray studies". *Nucleic Acids Research* 43.7 (2015), e47.

[233] Nueda, M. J., Tarazona, S., and Conesa, A. "Next maSigPro: Updating maSigPro bioconductor package for RNA-seq time series". *Bioinformatics* 30.18 (2014), pp. 2598–2602.

[234] Szklarczyk, D., Franceschini, A., Wyder, S., Forslund, K., Heller, D., Huerta-Cepas, J., Simonovic, M., Roth, A., Santos, A., Tsafou, K. P., Kuhn, M., Bork, P., Jensen, L. J., and Von Mering, C. "STRING v10: Protein-protein interaction networks, integrated over the tree of life". *Nucleic Acids Research* 43.D1 (2015), pp. D447–D452.

[235] Bader, G. D. and Hogue, C. W. "An automated method for finding molecular complexes in large protein interaction networks". *BMC Bioinformatics* 4.1 (2003), p. 2.

[236] Mi, H., Muruganujan, A., Casagrande, J. T., and Thomas, P. D. "Large-scale gene function analysis with the PANTHER classification system." *Nature protocols* 8.8 (2013), pp. 1551–1566.

[237] Jones, E., Oliphant, T., Peterson, P., et al. "SciPy: Open Source Scientific Tools for Python". 9 (2015), pp. 10–20.

[238] Oliphant, T. E. *A guide to NumPy*. Vol. 1. Trelgol Publishing USA, 2006.

[239] McKinney, W. "Data Structures for Statistical Computing in Python". *Proceedings of the 9th Python in Science Conference*. Vol. 1697900. Scipy. Austin, TX. 2010, pp. 51–56.

[240] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. "Scikit-learn: Machine Learning in Python". *Journal of Machine Learning Research* 12.Oct (2011), pp. 2825–2830.

[241] Hagberg, A. A., Schult, D. A., and Swart, P. J. *Exploring network structure, dynamics, and function using NetworkX*. Tech. rep. Los Alamos National Lab.(LANL), Los Alamos, NM (United States), 2008, pp. 11–15.

[242] Pérez, F. and Granger, B. E. "IPython: A system for interactive scientific computing". *Computing in Science and Engineering* 9.3 (2007), pp. 21–29.

[243] Kluyver, T., Ragan-kelley, B., Pérez, F., Granger, B., Bussonnier, M., Frederic, J., Kelley, K., Hamrick, J., Grout, J., Corlay, S., Ivanov, P., Avila, D., Abdalla, S., and Willing, C. "Jupyter Notebooks—a publishing format for reproducible computational workflows". *Positioning and Power in Academic Publishing: Players, Agents and Agendas*. 2016, pp. 87–90.

[244] Hunter, J. D. "Matplotlib: A 2D graphics environment". *Computing in Science and Engineering* 9.3 (2007), pp. 99–104.

[245] Brown, C. J., Kaufman, T., Trinidad, J. C., and Clemmer, D. E. "Proteome changes in the aging Drosophila melanogaster head". *International Journal of Mass Spectrometry* (2018).

[246] Tain, L. S., Sehlke, R., Jain, C., Chokkalingam, M., Nagaraj, N., Essers, P., Rassner, M., Grönke, S., Froelich, J., Dieterich, C., Mann, M., Alic, N., Beyer, A., and Partridge, L. "A proteomic atlas of insulin signalling reveals tissue-specific mechanisms of longevity assurance". *Molecular Systems Biology* (2017).

[247] Anders, S. and Huber, W. "Differential expression analysis for sequence count data". *Genome Biology* (2010).

[248] Zhang, Z. H., Jhaveri, D. J., Marshall, V. M., Bauer, D. C., Edson, J., Narayanan, R. K., Robinson, G. J., Lundberg, A. E., Bartlett, P. F., Wray, N. R., and Zhao, Q. Y. "A comparative study of techniques for differential expression analysis on RNA-seq data". *PLoS ONE* 9.8 (2014). Ed. by Provero, P., e103207.

[249] Seyednasrollah, F., Laiho, A., and Elo, L. L. "Comparison of software packages for detecting differential expression in RNA-seq studies". *Briefings in Bioinformatics* 16.1 (2013), pp. 59–70.

[250] Yu, H., Kim, P. M., Sprecher, E., Trifonov, V., and Gerstein, M. "The importance of bottlenecks in protein networks: Correlation with gene essentiality and expression dynamics". *PLoS Computational Biology* 3.4 (2007), e59.

[251] Savas, J. N., Wang, Y.-Z., DeNardo, L. A., Martinez-Bartolome, S., McClatchy, D. B., Hark, T. J., Shanks, N. F., Cozzolino, K. A., Lavallée-Adam, M., Smukowski, S. N., Park, S. K., Kelly, J. W., Koo, E. H., Nakagawa, T., Masliah, E., Ghosh, A., and Yates, J. R. "Amyloid Accumulation Drives Proteome-wide Alterations in Mouse Models of Alzheimer's Disease-like Pathology." *Cell reports* 21.9 (2017), pp. 2614–2627.

[252] Niccoli, T., Cabecinha, M., Tillmann, A., Kerr, F., Wong, C. T., Cardenes, D., Vincent, A. J., Bettedi, L., Li, L., Grönke, S., Dols, J., and Partridge, L. "Increased Glucose Transport into Neurons Rescues Aβ Toxicity in Drosophila". *Current Biology* (2016).

[253] Liu, C. C., Kanekiyo, T., Xu, H., and Bu, G. "Apolipoprotein e and Alzheimer disease: Risk, mechanisms and therapy". *Nature Reviews Neurology* (2013).

[254] Palm, W., Sampaio, J. L., Brankatschk, M., Carvalho, M., Mahmoud, A., Shevchenko, A., and Eaton, S. "Lipoproteins in Drosophila melanogaster-assembly, function, and influence on tissue lipid composition". *PLoS Genetics* (2012).

[255] Bereczki, E., Bernat, G., Csont, T., Ferdinandy, P., Scheich, H., and Sántha, M. "Overexpression of human apolipoprotein B-100 induces severe neurodegeneration in transgenic mice". *Journal of Proteome Research* (2008).

[256] Löffler, T., Flunkert, S., Havas, D., Sántha, M., Hutter-Paier, B., Steyrer, E., and Windisch, M. "Impact of ApoB-100 expression on cognition and brain pathology in wild-type and hAPPsl mice". *Neurobiology of Aging* 34.10 (2013), pp. 2379–2388.

[257] Caramelli, P., Nitrini, R., Maranhao, R., Lourenço, A. C., Damasceno, M. C., Vinagre, C., and Caramelli, B. "Increased apolipoprotein B serum concentration in Alzheimer's disease". *Acta Neurologica Scandinavica* (1999).

[258] Zhang, R., Barker, L., Pinchev, D., Marshall, J., Rasamoelisolo, M., Smith, C., Kupchak, P., Kireeva, I., Ingratta, L., and Jackowski, G. "Mining biomarkers in human sera using proteomic tools". *Proteomics* (2004).

[259] López-Otín, C., Blasco, M. A., Partridge, L., Serrano, M., and Kroemer, G. "The hallmarks of aging". *Cell* 153.6 (2013), pp. 1194–1217.

[260] Hou, Y., Dan, X., Babbar, M., Wei, Y., Hasselbalch, S. G., Croteau, D. L., and Bohr, V. A. "Ageing as a risk factor for neurodegenerative disease". *Nature Reviews Neurology* (2019).

[261] Afshordel, S., Wood, W. G., Igbavboa, U., Muller, W. E., and Eckert, G. P. "Impaired geranylgeranyltransferase-I regulation reduces membrane-associated Rho protein levels in aged mouse brain". *Journal of Neurochemistry* (2014).

[262] Gao, S., Yu, R., and Zhou, X. "The Role of Geranylgeranyltransferase I-Mediated Protein Prenylation in the Brain". *Molecular Neurobiology* (2016).

[263] D'Souza, Y., Elharram, A., Soon-Shiong, R., Andrew, R. D., and Bennett, B. M. "Characterization of Aldh2-/- mice as an age-related model of cognitive impairment and Alzheimer's disease". *Molecular Brain* (2015).

[264] Ohsawa, I., Nishimaki, K., Murakami, Y., Suzuki, Y., Ishikawa, M., and Ohta, S. "Age-dependent neurodegeneration accompanying memory loss in transgenic mice defective in mitochondrial aldehyde dehydrogenase 2 activity". *Journal of Neuroscience* (2008).

[265] Sade, Y., Toker, L., Kara, N. Z., Einat, H., Rapoport, S., Moechars, D., Berry, G. T., Bersudsky, Y., and Agam, G. "IP3 accumulation and/or inositol depletion: Two downstream lithium's effects that may mediate its behavioral and cellular changes". *Translational Psychiatry* (2016).

[266] Dobrin, S. E. and Fahrbach, S. E. "Rho GTPase activity in the honey bee mushroom bodies is correlated with age and foraging experience". *Journal of Insect Physiology* (2012).

[267] Owen, L. and Sunram-Lea, S. I. "Metabolic agents that enhance ATP can improve cognitive functioning: A review of the evidence for glucose, oxygen, pyruvate, creatine, and L-carnitine". *Nutrients* 3.8 (2011), pp. 735–755.

[268] Maynard, S., Fang, E. F., Scheibye-Knudsen, M., Croteau, D. L., and Bohr, V. A. "DNA damage, DNA repair, aging, and neurodegeneration". *Cold Spring Harbor Perspectives in Medicine* (2015).

[269] Anisimova, A. S., Alexandrov, A. I., Makarova, N. E., Gladyshev, V. N., and Dmitriev, S. E. "Protein synthesis and quality control in aging". *Aging* 10.12 (2018), pp. 4269–4288.

[270] Mattson, M. P. and Arumugam, T. V. "Hallmarks of Brain Aging: Adaptive and Pathological Modification by Metabolic States". *Cell Metabolism* 27.6 (2018), pp. 1176–1199.

[271] Maas, A. I. "Cerebrospinal fluid enzymes in acute brain injury 2 Relation of CSF enzyme activity to extent of brain injury". *Journal of Neurology, Neurosurgery and Psychiatry* 40.7 (1977), pp. 666–674.

[272] Casley, C. S., Canevari, L., Land, J. M., Clark, J. B., and Sharpe, M. A. "β-Amyloid inhibits integrated mitochondrial respiration and key enzyme activities". *Journal of Neurochemistry* (2002).

[273] Cardoso, S. M., Proença, M. T., Santos, S., Santana, I., and Oliveira, C. R. "Cytochrome c oxidase is decreased in Alzheimer's disease platelets". *Neurobiology of Aging* (2004).

[274] Fukui, H., Diaz, F., Garcia, S., and Moraes, C. T. "Cytochrome c oxidase deficiency in neurons decreases both oxidative stress and amyloid formation in a mouse model of Alzheimer's disease". *Proceedings of the National Academy of Sciences of the United States of America* (2007).

[275] Castellani, R., Siedlak, S., Fortino, A., Perry, G., Ghetti, B., and Smith, M. "Chitin-like Polysaccharides in Alzheimers Disease Brains". *Current Alzheimer Research* (2005).

[276] Kommaddi, R. P., Das, D., Karunakaran, S., Nanguneri, S., Bapat, D., Ray, A., Shaw, E., Bennett, D. A., Nair, D., and Ravindranath, V. "Aβ mediates F-actin disassembly in dendritic spines leading to cognitive deficits in alzheimer's disease". *Journal of Neuroscience* (2018).

[277] Hu, Y., Flockhart, I., Vinayagam, A., Bergwitz, C., Berger, B., Perrimon, N., and Mohr, S. E. "An integrative approach to ortholog prediction for disease-focused and other functional studies". *BMC Bioinformatics* (2011).

[278] Meloni, I., Muscettola, M., Raynaud, M., Longo, I., Bruttini, M., Moizard, M. P., Gomot, M., Chelly, J., Des Portes, V., Fryns, J. P., Ropers, H. H., Magi, B., Bellan, C., Volpi, N., Yntema, H. G., Lewis, S. E., Schaffer, J. E., and Renieri, A. "FACL4, encoding fatty acid-CoA ligase 4, is mutated in nonspecific X-linked mental retardation". *Nature Genetics* (2002).

[279] Peters, H., Buck, N., Wanders, R., Ruiter, J., Waterham, H., Koster, J., Yaplito-Lee, J., Ferdinandusse, S., and Pitt, J. "ECHS1 mutations in Leigh disease: A new inborn error of metabolism affecting valine metabolism". *Brain* (2014).

[280] Datta, A., Akatsu, H., Heese, K., and Sze, S. K. "Quantitative clinical proteomic study of autopsied human infarcted brain specimens to elucidate the deregulated pathways in ischemic stroke pathology". *Journal of Proteomics* (2013).

[281] McKenzie, A. T., Moyon, S., Wang, M., Katsyv, I., Song, W. M., Zhou, X., Dammer, E. B., Duong, D. M., Aaker, J., Zhao, Y., Beckmann, N., Wang, P., Zhu, J., Lah, J. J., Seyfried, N. T., Levey, A. I., Katsel, P., Haroutunian, V., Schadt, E. E., Popko, B., et al. "Multiscale network modeling of oligodendrocytes reveals molecular components of myelin dysregulation in Alzheimer's disease". *Molecular Neurodegeneration* (2017).

[282] Chi, L. M., Wang, X., and Nan, G. X. "In silico analyses for molecular genetic mechanism and candidate genes in patients with Alzheimer's disease". *Acta Neurologica Belgica* (2016).

[283] Gerber, H., Mosser, S., Boury-Jamot, B., Stumpe, M., Piersigilli, A., Goepfert, C., Dengjel, J., Albrecht, U., Magara, F., and Fraering, P. C. "The APMAP interactome reveals new modulators of APP processing and beta-amyloid production that are altered in Alzheimer's disease". *Acta neuropathologica communications* (2019).

[284] Terzioglu-Usak, S., Negis, Y., Karabulut, D. S., Zaim, M., and Isik, S. "Cellular Model of Alzheimer's Disease: Aβ1-42 Peptide Induces Amyloid Deposition and a Decrease in Topo Isomerase IIβ and Nurr1 Expression". *Current Alzheimer Research* (2017).

[285] Tzekov, R., Dawson, C., Orlando, M., Mouzon, B., Reed, J., Evans, J., Crynen, G., Mullan, M., and Crawford, F. "Sub-Chronic neuropathological and biochemical changes in mouse visual system after repetitive mild traumatic brain injury". *PLoS ONE* (2016).

[286] Matthias E, F., Ravi Kiran Reddy, K., Joaquin, G. L., Susana, M., and Kameshwar RS, A. "The unfolded protein response and its potential role in Huntington's disease elucidated by a systems biology approach". *F1000Research* (2015).

[287] Talwar, P., Silla, Y., Grover, S., Gupta, M., Agarwal, R., Kushwaha, S., and Kukreti, R. "Genomic convergence and network analysis approach to identify candidate genes in Alzheimer's disease". *BMC Genomics* (2014).

[288] Rogers, I., Kerr, F., Martinez, P., Hardy, J., Lovestone, S., and Partridge, L. "Ageing increases vulnerability to Aβ42 toxicity in Drosophila". *PLoS ONE* (2012).

[289] Jacobson, G. R. and Rosenbusch, J. P. "ATP binding to a protease resistant core of actin". *Proceedings of the National Academy of Sciences of the United States of America* (1976).

[290] Hozumi, T. "Structural aspects of skeletal muscle F-actin as studied by tryptic digestion: Evidence for a second nucleotide interacting site". *Journal of Biochemistry* (1988).

[291] Rodriguez-Suarez, E., Hughes, C., Gethings, L., Giles, K., Wildgoose, J., Stapels, M., E. Fadgen, K., J. Geromanos, S., P.C. Vissers, J., Elortza, F., and I. Langridge, J. "An Ion Mobility Assisted Data Independent LC-MS Strategy for the Analysis of Complex Biological Samples". *Current Analytical Chemistry* 9.2 (2013), pp. 199–211.

[292] Zhu, Y., Orre, L. M., Tran, Y. Z., Mermelekas, G., Johansson, H. J., Malyutina, A., Anders, S., and Lehtiö, J. "DEqMS: A method for accurate variance estimation in differential protein expression analysis". *Molecular and Cellular Proteomics* (2020).

[293] Zhang, X., Smits, A. H., Van Tilburg, G. B., Ovaa, H., Huber, W., and Vermeulen, M. "Proteome-wide identification of ubiquitin interactions using UbIA-MS". *Nature Protocols* (2018).

[294] Sberro, H., Fremin, B. J., Zlitni, S., Edfors, F., Greenfield, N., Snyder, M. P., Pavlopoulos, G. A., Kyrpides, N. C., and Bhatt, A. S. "Large-Scale Analyses of Human Microbiomes Reveal Thousands of Small, Novel Genes". *Cell* 178.5 (2019), 1245–1259.e14.

[295] Rinke, C., Schwientek, P., Sczyrba, A., Ivanova, N. N., Anderson, I. J., Cheng, J. F., Darling, A., Malfatti, S., Swan, B. K., Gies, E. A., Dodsworth, J. A., Hedlund, B. P., Tsiamis, G., Sievert, S. M., Liu, W. T., Eisen, J. A., Hallam, S. J., Kyrpides, N. C., Stepanauskas, R., Rubin, E. M., et al. "Insights into the phylogeny and coding potential of microbial dark matter". *Nature* (2013).

[296] Rappé, M. S. and Giovannoni, S. J. "The Uncultured Microbial Majority". *Annual Review of Microbiology* (2003).

[297] Saary, P., Mitchell, A. L., and Finn, R. D. "Estimating the quality of eukaryotic genomes recovered from metagenomic analysis". *bioRxiv* (2019), p. 2019.12.19.882753.

[298] Huttenhower, C., Gevers, D., Knight, R., Abubucker, S., Badger, J. H., Chinwalla, A. T., Creasy, H. H., Earl, A. M., Fitzgerald, M. G., Fulton, R. S., Giglio, M. G., Hallsworth-Pepin, K., Lobos, E. A., Madupu, R., Magrini, V., Martin, J. C., Mitreva, M., Muzny, D. M., Sodergren, E. J., Versalovic, J., et al. "Structure, function and diversity of the healthy human microbiome". *Nature* (2012).

[299] Almeida, A., Mitchell, A. L., Boland, M., Forster, S. C., Gloor, G. B., Tarkowska, A., Lawley, T. D., and Finn, R. D. "A new genomic blueprint of the human gut microbiota". *Nature* (2019).

[300] Forster, S. C., Kumar, N., Anonye, B. O., Almeida, A., Viciani, E., Stares, M. D., Dunn, M., Mkandawire, T. T., Zhu, A., Shao, Y., Pike, L. J., Louie, T., Browne, H. P., Mitchell, A. L., Neville, B. A., Finn, R. D., and Lawley, T. D. "A human gut bacterial genome and culture collection for improved metagenomic analyses". *Nature Biotechnology* (2019).

[301] Sunagawa, S., Coelho, L. P., Chaffron, S., Kultima, J. R., Labadie, K., Salazar, G., Djahanschiri, B., Zeller, G., Mende, D. R., Alberti, A., Cornejo-Castillo, F. M., Costea, P. I., Cruaud, C., D'Ovidio, F., Engelen, S., Ferrera, I., Gasol, J. M., Guidi, L., Hilde-

brand, F., Kokoszka, F., et al. “Structure and function of the global ocean microbiome”. *Science* (2015).

[302] Sunagawa, S., Acinas, S. G., Bork, P., Bowler, C., Eveillard, D., Gorsky, G., Guidi, L., Iudicone, D., Karsenti, E., Lombard, F., Ogata, H., Pesant, S., Sullivan, M. B., Wincker, P., and Vargas, C. de. “Tara Oceans: towards global ocean ecosystems biology”. *Nature Reviews Microbiology* (2020), pp. 1–18.

[303] Al-Shayeb, B., Sachdeva, R., Chen, L. X., Ward, F., Munk, P., Devoto, A., Castelle, C. J., Olm, M. R., Bouma-Gregson, K., Amano, Y., He, C., Méheust, R., Brooks, B., Thomas, A., Lavy, A., Matheus-Carnevali, P., Sun, C., Goltsman, D. S., Borton, M. A., Sharrar, A., et al. “Clades of huge phages from across Earth’s ecosystems”. *Nature* (2020).

[304] Rothschild, D., Weissbrod, O., Barkan, E., Kurilshikov, A., Korem, T., Zeevi, D., Costea, P. I., Godneva, A., Kalka, I. N., Bar, N., Shilo, S., Lador, D., Vila, A. V., Zmora, N., Pevsner-Fischer, M., Israeli, D., Kosower, N., Malka, G., Wolf, B. C., Avnit-Sagi, T., et al. “Environment dominates over host genetics in shaping human gut microbiota”. *Nature* 555.7695 (2018), pp. 210–215.

[305] Valles-Colomer, M., Falony, G., Darzi, Y., Tigchelaar, E. F., Wang, J., Tito, R. Y., Schiweck, C., Kurilshikov, A., Joossens, M., Wijmenga, C., Claes, S., Van Oudenhove, L., Zhernakova, A., Vieira-Silva, S., and Raes, J. “The neuroactive potential of the human gut microbiota in quality of life and depression”. *Nature Microbiology* (2019).

[306] Bachmann, N. L., Rockett, R. J., Timms, V. J., and Sintchenko, V. “Advances in clinical sample preparation for identification and characterization of bacterial pathogens using metagenomics”. *Frontiers in Public Health* 6.DEC (2018).

[307] Franzosa, E. A., Morgan, X. C., Segata, N., Waldron, L., Reyes, J., Earl, A. M., Giannoukos, G., Boylan, M. R., Ciulla, D., Gevers, D., Izard, J., Garrett, W. S., Chan, A. T., and Huttenhower, C. “Relating the metatranscriptome and metagenome of the human gut”. *Proceedings of the National Academy of Sciences of the United States of America* (2014).

[308] Bowers, R. M., Clum, A., Tice, H., Lim, J., Singh, K., Ciobanu, D., Ngan, C. Y., Cheng, J. F., Tringe, S. G., and Woyke, T. “Impact of library preparation protocols and template quantity on the metagenomic reconstruction of a mock microbial community”. *BMC Genomics* (2015).

[309] Lozupone, C. A., Stombaugh, J., Gonzalez, A., Ackermann, G., Wendel, D., Vázquez-Baeza, Y., Jansson, J. K., Gordon, J. I., and Knight, R. “Meta-analyses of studies of the human microbiota”. *Genome Research* (2013).

[310] Voigt, A. Y., Costea, P. I., Kultima, J. R., Li, S. S., Zeller, G., Sunagawa, S., and Bork, P. “Temporal and technical variability of human gut metagenomes”. *Genome Biology* (2015).

[311] Solonenko, S. A., Ignacio-Espinoza, J. C., Alberti, A., Cruaud, C., Hallam, S., Konstantinidis, K., Tyson, G., Wincker, P., and Sullivan, M. B. “Sequencing platform and library preparation choices impact viral metagenomes”. *BMC Genomics* (2013).

[312] Raes, J. and Bork, P. “Molecular eco-systems biology: Towards an understanding of community function”. *Nature Reviews Microbiology* (2008).

[313] Costea, P. I., Zeller, G., Sunagawa, S., Pelletier, E., Alberti, A., Levenez, F., Tramontano, M., Driessen, M., Hercog, R., Jung, F. E., Kultima, J. R., Hayward, M. R., Coelho, L. P., Allen-Vercoe, E., Bertrand, L., Blaut, M., Brown, J. R., Carton, T., Cools-Portier, S., Daigneault, M., et al. “Towards standards for human fecal sample processing in metagenomic studies”. *Nature Biotechnology* (2017).

[314] Kallies, R., Hölzer, M., Toscan, R. B., Rocha, U. N. da, Anders, J., Marz, M., and Chatzinotas, A. "Evaluation of sequencing library preparation protocols for viral metagenomic analysis from pristine aquifer groundwaters". *Viruses* (2019).

[315] Sato, M. P., Ogura, Y., Nakamura, K., Nishida, R., Gotoh, Y., Hayashi, M., Hisatsune, J., Sugai, M., Takehiko, I., and Hayashi, T. "Comparison of the sequencing bias of currently available library preparation kits for Illumina sequencing of bacterial genomes and metagenomes". *DNA Research* 26.5 (2019), pp. 391–398.

[316] Sevim, V., Lee, J., Egan, R., Clum, A., Hundley, H., Lee, J., Everroad, R. C., Detweiler, A. M., Bebout, B. M., Pett-Ridge, J., Göker, M., Murray, A. E., Lindemann, S. R., Klenk, H. P., O'Malley, R., Zane, M., Cheng, J. F., Copeland, A., Daum, C., Singer, E., and Woyke, T. "Shotgun metagenome data of a defined mock community using Oxford Nanopore, PacBio and Illumina technologies". *Scientific data* (2019).

[317] Rodrigue, S., Materna, A. C., Timberlake, S. C., Blackburn, M. C., Malmstrom, R. R., Alm, E. J., and Chisholm, S. W. "Unlocking short read sequencing for metagenomics". *PLoS ONE* (2010).

[318] Deamer, D., Akeson, M., and Branton, D. "Three decades of nanopore sequencing". *Nature Biotechnology* 34.5 (2016), pp. 518–524.

[319] Urban, L., Holzer, A., Baronas, J. J., Hall, M., Braeuninger-Weimer, P., Scherm, M. J., Kunz, D. J., Perera, S. N., Martin-Herranz, D. E., Tipper, E. T., Salter, S. J., and Stammnitz, M. R. "Freshwater monitoring by nanopore sequencing". *bioRxiv* (2020), p. 2020.02.06.936302.

[320] Jain, M., Olsen, H. E., Paten, B., and Akeson, M. "The Oxford Nanopore MinION: Delivery of nanopore sequencing to the genomics community". *Genome Biology* (2016).

[321] Church, G. and Deamer, D. W. *Characterization of individual polymer molecules based on monomer-interface interactions*. 1996.

[322] Ouldali, H., Sarthak, K., Ensslen, T., Piguet, F., Manivet, P., Pelta, J., Behrends, J. C., Aksimentiev, A., and Oukhaled, A. "Electrical recognition of the twenty proteinogenic amino acids using an aerolysin nanopore". *Nature Biotechnology* 38.2 (2020), pp. 176–181.

[323] Rang, F. J., Kloosterman, W. P., and Ridder, J. de. "From squiggle to basepair: Computational approaches for improving nanopore sequencing read accuracy". *Genome Biology* 19.1 (2018).

[324] Wick, R. R., Judd, L. M., and Holt, K. E. "Performance of neural network basecalling tools for Oxford Nanopore sequencing". *Genome Biology* (2019).

[325] Antipov, D., Korobeynikov, A., McLean, J. S., and Pevzner, P. A. "HybridSPAdes: An algorithm for hybrid assembly of short and long reads". *Bioinformatics* (2016).

[326] Overholt, W. A., Hölzer, M., Geesink, P., Diezel, C., Marz, M., and Küsel, K. "Inclusion of Oxford Nanopore long reads improves all microbial and phage metagenome-assembled genomes from a complex aquifer system". *bioRxiv* (2019).

[327] Nurk, S., Meleshko, D., Korobeynikov, A., and Pevzner, P. A. "MetaSPAdes: A new versatile metagenomic assembler". *Genome Research* 27.5 (2017), pp. 824–834.

[328] Steinegger, M. and Söding, J. "MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets". *Nature Biotechnology* 35.11 (2017), pp. 1026–1028.

[329] Sczyrba, A., Hofmann, P., Belmann, P., Koslicki, D., Janssen, S., Dröge, J., Gregor, I., Majda, S., Fiedler, J., Dahms, E., Bremges, A., Fritz, A., Garrido-Oter, R., Jørgensen, T. S., Shapiro, N., Blood, P. D., Gurevich, A., Bai, Y., Turaev, D., Demaere, M. Z., et

al. “Critical Assessment of Metagenome Interpretation - A benchmark of metagenomics software”. *Nature Methods* (2017).

[330] Mitchell, A. L., Almeida, A., Beracochea, M., Boland, M., Burgin, J., Cochrane, G., Crusoe, M. R., Kale, V., Potter, S. C., Richardson, L. J., Sakharova, E., Scheremetjew, M., Korobeynikov, A., Shlemov, A., Kunyavskaya, O., Lapidus, A., and Finn, R. D. “MGnify: the microbiome analysis resource in 2020”. *Nucleic acids research* (2020).

[331] Amstutz, P., Crusoe, M. R., Tijanić, N., Chapman, B., Chilton, J., Heuer, M., Kartashov, A., Leehr, D., Ménager, H., Nedeljkovich, M., Scales, M., Soiland-Reyes, S., and Stojanovic, L. “Common Workflow Language Specifications, v1.0”. *Figshare* (2016).

[332] Amid, C., Alako, B. T., Balavenkataraman Kadhirvelu, V., Burdett, T., Burgin, J., Fan, J., Harrison, P. W., Holt, S., Hussein, A., Ivanov, E., Jayathilaka, S., Kay, S., Keane, T., Leinonen, R., Liu, X., Martinez-Villacorta, J., Milano, A., Pakseresht, A., Rahman, N., Rajan, J., et al. “The European Nucleotide Archive in 2019”. *Nucleic acids research* (2020).

[333] Bruijn, N. G. “A combinatorial problem”. *Proc. Koninklijke Nederlandse Academie van Wetenschappen* 49 (1946), pp. 758–764.

[334] Myers, E. W. “Toward Simplifying and Accurately Formulating Fragment Assembly”. *Journal of Computational Biology* 2.2 (1995), pp. 275–290.

[335] Compeau, P. E., Pevzner, P. A., and Tesler, G. “How to apply de Bruijn graphs to genome assembly”. *Nature Biotechnology* 29.11 (2011), pp. 987–991.

[336] Euler, L. “Solutio problematis ad geometriam situs pertinentis” ().

[337] Garey, M. R. and Johnson, D. S. “Computers and Intractability: A Guide to the Theory of NP-Completeness (Series of Books in the Mathematical Sciences)”. *Computers and Intractability* (1979).

[338] Lantz, H., Dominguez Del Angel, V., Hjerde, E., Sterck, L., Capella-Gutierrez, S., Notredame, C., Vinnere Pettersson, O., Amselem, J., Bouri, L., Bocs, S., Klopp, C., Gibrat, J. F., Vlasova, A., Leskosek, B. L., Soler, L., and Binzer-Panchal, M. “Ten steps to get started in Genome Assembly and Annotation”. *F1000Research* 7 (2018).

[339] Hyatt, D., Chen, G. L., LoCascio, P. F., Land, M. L., Larimer, F. W., and Hauser, L. J. “Prodigal: Prokaryotic gene recognition and translation initiation site identification”. *BMC Bioinformatics* 11 (2010), p. 119.

[340] Ismail, W. M., Ye, Y., and Tang, H. “Gene finding in metatranscriptomic sequences”. *BMC Bioinformatics* 15.9 (2014).

[341] Sharon, I. and Banfield, J. F. “Genomes from metagenomics”. *Science* 342.6162 (2013), pp. 1057–1058.

[342] Tyson, G. W., Chapman, J., Hugenholtz, P., Allen, E. E., Ram, R. J., Richardson, P. M., Solovyev, V. V., Rubin, E. M., Rokhsar, D. S., and Banfield, J. F. “Community structure and metabolism through reconstruction of microbial genomes from the environment”. *Nature* 428.6978 (2004), pp. 37–43.

[343] Parks, D. H., Rinke, C., Chuvochina, M., Chaumeil, P. A., Woodcroft, B. J., Evans, P. N., Hugenholtz, P., and Tyson, G. W. “Recovery of nearly 8,000 metagenome-assembled genomes substantially expands the tree of life”. *Nature Microbiology* (2017).

[344] Bowers, R. M., Kyrpides, N. C., Stepanauskas, R., Harmon-Smith, M., Doud, D., Reddy, T. B., Schulz, F., Jarett, J., Rivers, A. R., Eloe-Fadrosh, E. A., Tringe, S. G., Ivanova, N. N., Copeland, A., Clum, A., Becraft, E. D., Malmstrom, R. R., Birren, B., Podar, M., Bork, P., Weinstock, G. M., et al. “Minimum information about a sin-

gle amplified genome (MISAG) and a metagenome-assembled genome (MIMAG) of bacteria and archaea". *Nature Biotechnology* 35.8 (2017), pp. 725–731.

[345] Kang, D. D., Froula, J., Egan, R., and Wang, Z. "MetaBAT, an efficient tool for accurately reconstructing single genomes from complex microbial communities". *PeerJ* 2015.8 (2015), e1165.

[346] Albertsen, M., Hugenholtz, P., Skarshewski, A., Nielsen, K. L., Tyson, G. W., and Nielsen, P. H. "Genome sequences of rare, uncultured bacteria obtained by differential coverage binning of multiple metagenomes". *Nature Biotechnology* 31.6 (2013), pp. 533–538.

[347] Ewels, P. A., Peltzer, A., Fillinger, S., Patel, H., Alneberg, J., Wilm, A., Garcia, M. U., Di Tommaso, P., and Nahnsen, S. "The nf-core framework for community-curated bioinformatics pipelines". *Nature Biotechnology* 38.3 (2020), pp. 276–278.

[348] Chaumeil, P.-A., Mussig, A. J., Hugenholtz, P., and Parks, D. H. "GTDB-Tk: a toolkit to classify genomes with the Genome Taxonomy Database". *Bioinformatics* (2019).

[349] Parks, D. H., Imelfort, M., Skennerton, C. T., Hugenholtz, P., and Tyson, G. W. "CheckM: Assessing the quality of microbial genomes recovered from isolates, single cells, and metagenomes". *Genome Research* (2015).

[350] Seppey, M., Manni, M., and Zdobnov, E. M. "BUSCO: Assessing genome assembly and annotation completeness". *Methods in Molecular Biology*. Vol. 1962. Humana Press Inc., 2019, pp. 227–245.

[351] Ollis, D. L., Cheah, E., Cygler, M., Dijkstra, B., Frolow, F., Franken, S. M., Harel, M., Remington, S. J., Silman, I., and Schrag, J. "The alpha/beta hydrolase fold." *Protein engineering* 5.3 (1992), pp. 197–211.

[352] Matthews, B. W., Sigler, P. B., Henderson, R., and Blow, D. M. "Three-dimensional structure of tosyl-α-chymotrypsin". *Nature* 214.5089 (1967), pp. 652–656.

[353] Dessailly, B. H., Redfern, O. C., Cuff, A. L., and Orengo, C. A. "Detailed Analysis of Function Divergence in a Large and Diverse Domain Superfamily: Toward a Refined Protocol of Function Classification". *Structure* (2010).

[354] Suplatov, D., Besenmatter, W., Švedas, V., and Svendsen, A. "Bioinformatic analysis of alpha/beta-hydrolase fold enzymes reveals subfamily-specific positions responsible for discrimination of amidase and lipase activities". *Protein Engineering, Design and Selection* 25.11 (2012), pp. 689–697.

[355] Mindrebo, J. T., Nartey, C. M., Seto, Y., Burkart, M. D., and Noel, J. P. "Unveiling the functional diversity of the alpha/beta hydrolase superfamily in the plant kingdom". *Current Opinion in Structural Biology* 41 (2016), pp. 233–246.

[356] Holmquist, M. "Alpha Beta-Hydrolase Fold Enzymes Structures, Functions and Mechanisms". *Current Protein and Peptide Science* 1.2 (2000), pp. 209–235.

[357] Zheng, Q., Wang, S., Duan, P., Liao, R., Chen, D., and Liu, W. "An α/β-hydrolase fold protein in the biosynthesis of thiostrepton exhibits a dual activity for endopeptidyl hydrolysis and epoxide ring opening/macrocyclization". *Proceedings of the National Academy of Sciences of the United States of America* 113.50 (2016), pp. 14318–14323.

[358] Lazniewski, M., Steczkiewicz, K., Knizewski, L., Wawer, I., and Ginalski, K. "Novel transmembrane lipases of alpha/beta hydrolase fold". *FEBS Letters* (2011).

[359] Rao, S. T. and Rossmann, M. G. "Comparison of super-secondary structures in proteins." *Journal of molecular biology* 76.2 (1973), pp. 241–56.

[360] Bateman, A. "UniProt: A worldwide hub of protein knowledge". *Nucleic Acids Research* (2019).

[361] Geyer, R., Jambeck, J. R., and Law, K. L. "Production, use, and fate of all plastics ever made". *Science Advances* 3.7 (2017), e1700782.

[362] Ragaert, K., Delva, L., and Van Geem, K. "Mechanical and chemical recycling of solid plastic waste". *Waste Management* 69 (2017), pp. 24–58.

[363] Bornscheuer, U. T. "Feeding on plastic". *Science* 351.6278 (2016), pp. 1154–1155.

[364] Wilcox, C., Van Sebille, E., Hardesty, B. D., and Estes, J. A. "Threat of plastic pollution to seabirds is global, pervasive, and increasing". *Proceedings of the National Academy of Sciences of the United States of America* 112.38 (2015), pp. 11899–11904.

[365] Law, K. L. and Thompson, R. C. "Microplastics in the seas". *Science* 345.6193 (2014), pp. 144–145.

[366] Rochman, C. M. "Microplastics research-from sink to source". *Science* 360.6384 (2018), pp. 28–29.

[367] Lebreton, L., Slat, B., Ferrari, F., Sainte-Rose, B., Aitken, J., Marthouse, R., Hajbane, S., Cunsolo, S., Schwarz, A., Levivier, A., Noble, K., Debeljak, P., Maral, H., Schoeneich-Argent, R., Brambini, R., and Reisser, J. "Evidence that the Great Pacific Garbage Patch is rapidly accumulating plastic". *Scientific Reports* 8.1 (2018), pp. 1–15.

[368] Lacerda, A. L., Rodrigues, L. d. S., Sebille, E. van, Rodrigues, F. L., Ribeiro, L., Secchi, E. R., Kessler, F., and Proietti, M. C. "Plastics in sea surface waters around the Antarctic Peninsula". *Scientific Reports* 9.1 (2019), pp. 1–12.

[369] Sharon, C. and Sharon, M. "Studies on biodegradation of polyethylene terephthalate: A synthetic polymer". *Journal of Microbiology and Biotechnology Research* (2013).

[370] Müller, R.-J., Schrader, H., Profe, J., Dresler, K., and Deckwer, W.-D. "Enzymatic Degradation of Poly(ethylene terephthalate): Rapid Hydrolyse using a Hydrolase fromT. fusca". *Macromolecular Rapid Communications* 26.17 (2005), pp. 1400–1405.

[371] Ronkvist, Å. M., Xie, W., Lu, W., and Gross, R. A. "Cutinase-Catalyzed hydrolysis of poly(ethylene terephthalate)". *Macromolecules* 42.14 (2009), pp. 5128–5138.

[372] Vertommen, M. A., Nierstrasz, V. A., Veer, M. V. D., and Warmoeskerken, M. M. "Enzymatic surface modification of poly(ethylene terephthalate)". *Journal of Biotechnology* 120.4 (2005), pp. 376–386.

[373] Gan, Z. and Zhang, H. "PMBD: a Comprehensive Plastics Microbial Biodegradation Database". *Database* 2019 (2019).

[374] Yoshida, S., Hiraga, K., Takehana, T., Taniguchi, I., Yamaji, H., Maeda, Y., Toyohara, K., Miyamoto, K., Kimura, Y., and Oda, K. "A bacterium that degrades and assimilates poly(ethylene terephthalate)". *Science* (2016).

[375] Yang, Y., Yang, J., and Jiang, L. "Comment on "A bacterium that degrades and assimilates poly(ethylene terephthalate)"". *Science* 353.6301 (2016), pp. 759–759.

[376] Yoshida, S., Hiraga, K., Takehana, T., Taniguchi, I., Yamaji, H., Maeda, Y., Toyohara, K., Miyamoto, K., Kimura, Y., and Oda, K. "Response to Comment on "a bacterium that degrades and assimilates poly(ethylene terephthalate)"". *Science* 353.6301 (2016), pp. 759–759.

[377] Han, X., Liu, W., Huang, J. W., Ma, J., Zheng, Y., Ko, T. P., Xu, L., Cheng, Y. S., Chen, C. C., and Guo, R. T. "Structural insight into catalytic mechanism of PET hydrolase". *Nature Communications* 8.1 (2017), pp. 1–6.

[378] Joo, S., Cho, I. J., Seo, H., Son, H. F., Sagong, H. Y., Shin, T. J., Choi, S. Y., Lee, S. Y., and Kim, K. J. "Structural insight into molecular mechanism of poly(ethylene terephthalate) degradation". *Nature Communications* 9.1 (2018), pp. 1–12.

[379] Austin, H. P., Allen, M. D., Donohoe, B. S., Rorrer, N. A., Kearns, F. L., Silveira, R. L., Pollard, B. C., Dominick, G., Duman, R., El Omari, K., Mykhaylyk, V., Wagner, A., Michener, W. E., Amore, A., Skaf, M. S., Crowley, M. F., Thorne, A. W., Johnson, C. W., Woodcock, H. L., McGeehan, J. E., and Beckham, G. T. "Characterization and engineering of a plastic-degrading aromatic polyesterase". *Proceedings of the National Academy of Sciences* 115.19 (2018), E4350–E4357.

[380] Palm, G. J., Reisky, L., Böttcher, D., Müller, H., Michels, E. A., Walczak, M. C., Berndt, L., Weiss, M. S., Bornscheuer, U. T., and Weber, G. "Structure of the plastic-degrading Ideonella sakaiensis MHETase bound to a substrate". *Nature Communications* 10.1 (2019), pp. 1–10.

[381] Tournier, V., Topham, C. M., Gilles, A., David, B., Folgoas, C., Moya-Leclair, E., Kamionka, E., Desrousseaux, M.-L., Texier, H., Gavalda, S., Cot, M., Guémard, E., Dalibey, M., Nomme, J., Cioci, G., Barbe, S., Chateau, M., André, I., Duquesne, S., and Marty, A. "An engineered PET depolymerase to break down and recycle plastic bottles". *Nature* 580.7802 (2020), pp. 216–219.

[382] Quince, C., Walker, A. W., Simpson, J. T., Loman, N. J., and Segata, N. "Shotgun metagenomics, from sampling to analysis". *Nature Biotechnology* 35.9 (2017), pp. 833–844.

[383] Kalvari, I., Argasinska, J., Quinones-Olvera, N., Nawrocki, E. P., Rivas, E., Eddy, S. R., Bateman, A., Finn, R. D., and Petrov, A. I. "Rfam 13.0: shifting to a genome-centric resource for non-coding RNA families." *Nucleic acids research* 46.D1 (2018), pp. D335–D342.

[384] Mukherjee, S., Stamatis, D., Bertsch, J., Ovchinnikova, G., Katta, H. Y., Mojica, A., Chen, I. M. A., Kyrpides, N. C., and Reddy, T. B. "Genomes OnLine database (GOLD) v.7: Updates and new features". *Nucleic Acids Research* (2019).

[385] Kurtzer, G. M., Sochat, V., and Bauer, M. W. "Singularity: Scientific containers for mobility of compute". *PLOS ONE* 12.5 (2017). Ed. by Gursoy, A., e0177459.

[386] da Veiga Leprevost, F., Grüning, B. A., Alves Aflitos, S., Röst, H. L., Uszkoreit, J., Barsnes, H., Vaudel, M., Moreno, P., Gatto, L., Weber, J., Bai, M., Jimenez, R. C., Sachsenberg, T., Pfeuffer, J., Vera Alvarez, R., Griss, J., Nesvizhskii, A. I., and Perez-Riverol, Y. "BioContainers: an open-source and community-driven framework for software standardization". *Bioinformatics* (2017).

[387] McCallum, A., Nigam, K., and Ungar, L. H. "Efficient clustering of high-dimensional data sets with application to reference matching". *Proceeding of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2000.

[388] Broder, A. Z., Charikar, M., Frieze, A. M., and Mitzenmacher, M. "Min-wise independent permutations (extended abstract)". *Proceedings of the thirtieth annual ACM symposium on Theory of computing - STOC '98*. New York, New York, USA: Association for Computing Machinery (ACM), 1998, pp. 327–336.

[389] Broder, A. Z. "On the resemblance and containment of documents". *Proceedings of the International Conference on Compression and Complexity of Sequences*. IEEE, 1997, pp. 21–29.

[390] DI Tommaso, P., Chatzou, M., Floden, E. W., Barja, P. P., Palumbo, E., and Notredame, C. "Nextflow enables reproducible computational workflows". *Nature Biotechnology* 35.4 (2017), pp. 316–319.

[391] Bezanson, J., Karpinski, S., Shah, V. B., and Edelman, A. "Julia: A Fast Dynamic Language for Technical Computing" (2012).

[392] Scholes, H. *harryscholes/CATHBase.jl: v0.1.0*. 2020.

[393] Nightingale, A., Antunes, R., Alpi, E., Bursteinas, B., Gonzales, L., Liu, W., Luo, J., Qi, G., Turner, E., and Martin, M. "The Proteins API: accessing key integrated protein and genome information". *Nucleic Acids Research* 45.W1 (2017), W539–W544.

[394] Henikoff, S. and Henikoff, J. G. "Amino acid substitution matrices from protein blocks". *Proceedings of the National Academy of Sciences of the United States of America* 89.22 (1992), pp. 10915–10919.

[395] Katoh, K. "MAFFT: a novel method for rapid multiple sequence alignment based on fast Fourier transform". *Nucleic Acids Research* (2002).

[396] Müllner, D. "Modern hierarchical, agglomerative clustering algorithms" (2011).

[397] Hie, B., Cho, H., DeMeo, B., Bryson, B., and Berger, B. "Geometric Sketching Compactly Summarizes the Single-Cell Transcriptomic Landscape". *Cell Systems* (2019).

[398] Rentzsch, R. and Orengo, C. A. "Protein function prediction - the power of multiplicity". *Trends in Biotechnology* 27.4 (2009), pp. 210–219.

[399] Bruna, T., Lomsadze, A., and Borodovsky, M. "GeneMark-EP and -EP+: automatic eukaryotic gene prediction supported by spliced aligned proteins". *bioRxiv* 17 (2020), p. 2019.12.31.891218.

[400] Levy Karin, E., Mirdita, M., and Söding, J. "MetaEuk—sensitive, high-throughput gene discovery, and annotation for large-scale eukaryotic metagenomics". *Microbiome* 8.1 (2020), p. 48.

[401] Sallet, E., Gouzy, J., and Schiex, T. "EuGene: An automated integrative gene finder for eukaryotes and prokaryotes". *Methods in Molecular Biology*. Vol. 1962. Humana Press Inc., 2019, pp. 97–120.

[402] Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T., Harris, M. A., Hill, D. P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J. C., Richardson, J. E., Ringwald, M., Rubin, G. M., Sherlock, G., and Consortium, G. O. "Gene Ontology: Tool for The Unification of Biology". *Nature Genetics* 25.1 (2000), pp. 25–29.

[403] Peled, S., Leiderman, O., Charar, R., Efroni, G., Shav-Tal, Y., and Ofran, Y. "De-novo protein function prediction using DNA binding and RNA binding proteins as a test case." *Nature communications* 7 (2016), p. 13424.

[404] Gillis, J. and Pavlidis, P. ""Guilt by association" is the exception rather than the rule in gene networks". *PLoS Computational Biology* 8.3 (2012). Ed. by Rzhetsky, A., e1002444.

[405] Szklarczyk, D., Morris, J. H., Cook, H., Kuhn, M., Wyder, S., Simonovic, M., Santos, A., Doncheva, N. T., Roth, A., Bork, P., Jensen, L. J., and Von Mering, C. "The STRING database in 2017: Quality-controlled protein-protein association networks, made broadly accessible". *Nucleic Acids Research* 45.D1 (2017), pp. D362–D368.

[406] Franceschini, A., Szklarczyk, D., Frankild, S., Kuhn, M., Simonovic, M., Roth, A., Lin, J., Minguez, P., Bork, P., Von Mering, C., and Jensen, L. J. "STRING v9.1: Protein-protein interaction networks, with increased coverage and integration". *Nucleic Acids Research* (2013).

[407] Chatr-Aryamontri, A., Oughtred, R., Boucher, L., Rust, J., Chang, C., Kolas, N. K., O'Donnell, L., Oster, S., Theesfeld, C., Sellam, A., Stark, C., Breitkreutz, B. J., Dolinski, K., and Tyers, M. "The BioGRID interaction database: 2017 update". *Nucleic Acids Research* 45.D1 (2017), pp. D369–D379.

[408] Pancaldi, V., Schubert, F., and Bähler, J. "Meta-analysis of genome regulation and expression variability across hundreds of environmental and genetic perturbations in fission yeast." *Molecular BioSystems* 6.3 (2010), pp. 543–52.

[409] Harris, M. A., Lock, A., Bähler, J., Oliver, S. G., and Wood, V. "FYPO: The fission yeast phenotype ontology". *Bioinformatics* 29.13 (2013), pp. 1671–1678.

[410] *Python*. URL: `https://www.python.org/` (visited on 03/14/2016).

[411] Chollet, F. et al. *Keras*. 2015.

[412] GoogleResearch. "TensorFlow: Large-scale machine learning on heterogeneous systems". *Google Research* (2015).

[413] Bähler, J. and Wood, V. "Probably the best model organism in the world". *Yeast* 23.13 (2006), pp. 899–900.

[414] Jeffares, D. C., Rallis, C., Rieux, A., Speed, D., Převorovský, M., Mourier, T., Marsellach, F. X., Iqbal, Z., Lau, W., Cheng, T. M. K., Pracana, R., Mülleder, M., Lawson, J. L. D., Chessel, A., Bala, S., Hellenthal, G., O'Fallon, B., Keane, T., Simpson, J. T., Bischof, L., et al. "The genomic and phenotypic diversity of Schizosaccharomyces pombe". *Nature Genetics* 47.3 (2015), pp. 235–241.

[415] Goffeau, A., Barrell, G., Bussey, H., Davis, R. W., Dujon, B., Feldmann, H., Galibert, F., Hoheisel, J. D., Jacq, C., Johnston, M., Louis, E. J., Mewes, H. W., Murakami, Y., Philippsen, P., Tettelin, H., and Oliver, S. G. "Life with 6000 genes". *Science* 274.5287 (1996), pp. 546–567.

[416] Hoffman, C. S., Wood, V., and Fantes, P. A. "An ancient yeast for young geneticists: A primer on the Schizosaccharomyces pombe model system". *Genetics* 201.2 (2015), pp. 403–423.

[417] Lee, M. G. "Molecular biology of the fission yeast". *Trends in Genetics* 6 (1990), p. 270.

[418] Wood, V., Gwilliam, R., Rajandream, M. A., Lyne, M., Lyne, R., Stewart, A., Sgouros, J., Peat, N., Hayles, J., Baker, S., Basham, D., Bowman, S., Brooks, K., Brown, D., Brown, S., Chillingworth, T., Churcher, C., Collins, M., Connor, R., Cronin, A., et al. "The genome sequence of Schizosaccharomyces pombe". *Nature* 415.6874 (2002), pp. 871–880.

[419] Sipiczki, M. "Where does fission yeast sit on the tree of life?" *Genome biology* 1.2 (2000), reviews1011.1.

[420] Nurse, P., Thuriaux, P., and Nasmyth, K. "Genetic control of the cell division cycle in the fission yeast Schizosaccharomyces pombe". *MGG Molecular & General Genetics* 146.2 (1976), pp. 167–178.

[421] Kim, D.-U., Hayles, J., Kim, D., Wood, V., Park, H.-O., Won, M., Yoo, H.-S., Duhig, T., Nam, M., Palmer, G., Han, S., Jeffery, L., Baek, S.-T., Lee, H., Shim, Y. S., Lee, M., Kim, L., Heo, K.-S., Noh, E. J., Lee, A.-R., et al. "Analysis of a genome-wide set of gene deletions in the fission yeast Schizosaccharomyces pombe". *Nature Biotechnology* 28.6 (2010), pp. 617–623.

[422] Han, T. X., Xu, X. Y., Zhang, M. J., Peng, X., and Du, L. L. "Global fitness profiling of fission yeast deletion strains by barcode sequencing". *Genome Biology* 11.6 (2010), R60.

[423] Smith, A. M., Heisler, L. E., Mellor, J., Kaper, F., Thompson, M. J., Chee, M., Roth, F. P., Giaever, G., and Nislow, C. "Quantitative phenotyping via deep barcode sequencing". *Genome Research* 19.10 (2009), pp. 1836–1842.

[424] Wood, V., Lock, A., Harris, M. A., Rutherford, K., Bähler, J., and Oliver, S. G. "Hidden in plain sight: What remains to be discovered in the eukaryotic proteome?" *Open Biology* (2019).

[425] Stoeger, T., Gerlach, M., Morimoto, R. I., and Nunes Amaral, L. A. "Large-scale investigation of the reasons why potentially important genes are ignored". *PLOS Biology* 16.9 (2018). Ed. by Freeman, T., e2006643.

[426] Lock, A., Rutherford, K., Harris, M. A., and Wood, V. "PomBase: The scientific resource for fission yeast". *Methods in Molecular Biology*. 2018.

[427] Hillenmeyer, M. E., Fung, E., Wildenhain, J., Pierce, S. E., Hoon, S., Lee, W., Proctor, M., St.Onge, R. P., Tyers, M., Koller, D., Altman, R. B., Davis, R. W., Nislow, C., and Giaever, G. "The Chemical Genomic Portrait of Yeast: Uncovering a Phenotype for All Genes". *Science* 320.5874 (2008), pp. 362–365.

[428] Schnoes, A. M., Ream, D. C., Thorman, A. W., Babbitt, P. C., and Friedberg, I. "Biases in the Experimental Annotations of Protein Function and Their Effect on Our Understanding of Protein Function Space". *PLoS Computational Biology* 9.5 (2013). Ed. by Orengo, C. A., e1003063.

[429] Saraç, Ö. S., Pancaldi, V., Bähler, J., and Beyer, A. "Topology of functional networks predicts physical binding of proteins". *Bioinformatics* (2012).

[430] Pancaldi, V., Saraç, O. S., Rallis, C., McLean, J. R., Převorovský, M., Gould, K., Beyer, A., and Bähler, J. "Predicting the fission yeast protein interaction network." *G3 (Bethesda, Md.)* 2.4 (2012), pp. 453–67.

[431] Lees, J. G., Hériché, J. K., Morilla, I., Fernández, J. M., Adler, P., Krallinger, M., Vilo, J., Valencia, A., Ellenberg, J., Ranea, J. A., and Orengo, C. "FUN-L: Gene prioritization for RNAi screens". *Bioinformatics* 31.12 (2015), pp. 2052–2053.

[432] Houle, D., Govindaraju, D. R., and Omholt, S. "Phenomics: The next challenge". *Nature Reviews Genetics* 11.12 (2010), pp. 855–866.

[433] Shuler, M. L. "Functional genomics: An opportunity for bioengineers". *Biotechnology Progress* 15.3 (1999), p. 287.

[434] Schilling, C. H., Edwards, J. S., and Palsson, B. O. "Toward metabolic phenomics: Analysis of genomic data using flux balances". *Biotechnology Progress* 15.3 (1999), pp. 288–295.

[435] Dove, A. "Proteomics: Translating genomics into products?" *Nature Biotechnology* 17.3 (1999), pp. 233–236.

[436] Mendel, G. "Experiments in plant hybridisation". *Verhandlungen des naturforschenden Ver-eines in Brünn* 4 (1866), pp. 3–47.

[437] Houle, D. "Numbering the hairs on our heads: The shared challenge and promise of phenomics". *Proceedings of the National Academy of Sciences of the United States of America* 107.SUPPL. 1 (2010), pp. 1793–1799.

[438] Novikoff, A. B. "The concept of integrative levels and biology". *Science* 101.2618 (1945), pp. 209–215.

[439] Murren, C. J. "The integrated phenotype". *Integrative and Comparative Biology*. 2012.

[440] Hoyles, L., Fernández-Real, J. M., Federici, M., Serino, M., Abbott, J., Charpentier, J., Heymes, C., Luque, J. L., Anthony, E., Barton, R. H., Chilloux, J., Myridakis, A., Martinez-Gili, L., Moreno-Navarrete, J. M., Benhamed, F., Azalbert, V., Blasco-Baque, V., Puig, J., Xifra, G., Ricart, W., et al. "Molecular phenomics and metagenomics of hepatic steatosis in non-diabetic obese women". *Nature Medicine* 24.7 (2018), pp. 1070–1080.

[441] Zhao, C., Zhang, Y., Du, J., Guo, X., Wen, W., Gu, S., Wang, J., and Fan, J. "Crop phenomics: Current status and perspectives". *Frontiers in Plant Science* 10 (2019), p. 714.

[442] Watt, M., Fiorani, F., Usadel, B., Rascher, U., Muller, O., and Schurr, U. "Phenotyping: New Windows into the Plant for Breeders". *Annual Review of Plant Biology* 71 (2020), pp. 689–712.

[443] Brown, S. D. M., Holmes, C. C., Mallon, A.-M., Meehan, T. F., Smedley, D., and Wells, S. "High-throughput mouse phenomics for characterizing mammalian gene function". *Nature Reviews Genetics* 19.6 (2018), pp. 357–370.

[444] Mali, P., Yang, L., Esvelt, K. M., Aach, J., Guell, M., DiCarlo, J. E., Norville, J. E., and Church, G. M. "RNA-guided human genome engineering via Cas9". *Science* 339.6121 (2013), pp. 823–826.

[445] Cong, L., Ran, F. A., Cox, D., Lin, S., Barretto, R., Habib, N., Hsu, P. D., Wu, X., Jiang, W., Marraffini, L. A., and Zhang, F. "Multiplex genome engineering using CRISPR/Cas systems". *Science* 339.6121 (2013), pp. 819–823.

[446] Gerlai, R. "Phenomics: Fiction or the future?" *Trends in Neurosciences* 25.10 (2002), pp. 506–509.

[447] Rallis, C. and Bähler, J. "Cell-based screens and phenomics with fission yeast". *Critical Reviews in Biochemistry and Molecular Biology* 51.2 (2016), pp. 86–95.

[448] Bischof, L., Převorovský, M., Rallis, C., Jeffares, D. C., Arzhaeva, Y., and Bähler, J. "Spotsizer: High-throughput quantitative analysis of microbial growth". *BioTechniques* 61.4 (2016), pp. 191–201.

[449] Zackrisson, M., Hallin, J., Ottosson, L.-G., Dahl, P., Fernandez-Parada, E., Ländström, E., Fernandez-Ricaud, L., Kaferle, P., Skyman, A., Stenberg, S., Omholt, S., Petrovič, U., Warringer, J., and Blomberg, A. "Scan-o-matic: High-Resolution Microbial Phenomics at a Massive Scale." *G3 (Bethesda, Md.)* 6.9 (2016), pp. 3003–14.

[450] Kamrad, S., Rodríguez-López, M., Cotobal, C., Correia-Melo, C., Ralser, M., and Bähler, J. "Pyphe: A python toolbox for assessing microbial growth and cell viability in high-throughput colony screens". *bioRxiv* (2020), p. 2020.01.22.915363.

[451] Sailem, H. Z., Rittscher, J., and Pelkmans, L. "KCML: a machine-learning framework for inference of multi-scale gene functions from genetic perturbation screens". *Molecular Systems Biology* 16.3 (2020).

[452] Romila, C.-A. "High-Throughput Chronological Lifespan Screening of the Fission Yeast Deletion Library Using Barcode Sequencing". *Doctoral thesis, UCL (University College London).* (2019).

[453] Rallis, C., Lopez-Maury, L., Georgescu, T., Pancaldi, V., and Bahler, J. "Systematic screen for mutants resistant to TORC1 inhibition in fission yeast reveals genes involved in cellular ageing and growth". *Biology Open* 3.2 (2014), pp. 161–171.

[454] Gentner, N. E. and Werner, M. M. "Repair in Schizosaccharomyces pombe as measured by recovery from caffeine enhancement of radiation-induced lethality". *MGG Molecular & General Genetics* 142.3 (1975), pp. 171–183.

[455] Osman, F. and McCready, S. "Differential effects of caffeine on DNA damage and replication cell cycle checkpoints in the fission yeast Schizosaccharomyces pombe". *Molecular and General Genetics* 260.4 (1998), pp. 319–334.

[456] Calvo, I. A., Gabrielli, N., Iglesias-Baena, I., García-Santamarina, S., Hoe, K.-L., Kim, D. U., Sansó, M., Zuin, A., Pérez, P., Ayté, J., and Hidalgo, E. "Genome-Wide Screen of Genes Required for Caffeine Tolerance in Fission Yeast". *PLoS ONE* 4.8 (2009). Ed. by Bonini, M., e6619.

[457] Zhou, H., Liu, Q., Shi, T., Yu, Y., and Lu, H. "Genome-wide screen of fission yeast mutants for sensitivity to 6-azauracil, an inhibitor of transcriptional elongation". *Yeast* 32.10 (2015), pp. 643–655.

[458] Noda, T. "Viability Assays to Monitor Yeast Autophagy". *Methods in enzymology*. Vol. 451. Academic Press, 2008, pp. 27–32.

[459] Benjamini, Y. and Hochberg, Y. "Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing". *Journal of the Royal Statistical Society B* 57.1 (1995), pp. 289–300.

[460] Dunn, O. J. "Multiple Comparisons among Means". *Journal of the American Statistical Association* 56.293 (1961), pp. 52–64.

[461] Dunn, O. J. "Estimation of the Medians for Dependent Variables". *The Annals of Mathematical Statistics* 30.1 (1959), pp. 192–197.

[462] The Gene Ontology Consortium. "Gene ontology consortium: Going forward". *Nucleic Acids Research* 43.D1 (2015), pp. D1049–D1056.

[463] Martignon, L., Vitouch, O., Takezawa, M., and Forster, M. R. "Naive and yet Enlightened: From Natural Frequencies to Fast and Frugal Decision Trees". *Thinking: Psychological Perspectives on Reasoning, Judgment and Decision Making*. John Wiley & Sons, Ltd, 2005, pp. 189–211.

[464] Raab, M. and Gigerenzer, G. "The power of simplicity: a fast-and-frugal heuristics approach to performance science". *Frontiers in Psychology* 6.OCT (2015), p. 1672.

[465] Murphy, K. P. *Machine learning : a probabilistic perspective*. MIT Press, 2012, p. 1067.

[466] Szklarczyk, D., Gable, A. L., Lyon, D., Junge, A., Wyder, S., Huerta-Cepas, J., Simonovic, M., Doncheva, N. T., Morris, J. H., Bork, P., Jensen, L. J., and Mering, C. von. "STRING v11: protein–protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets". *Nucleic Acids Research* 47.D1 (2019), pp. D607–D613.

[467] D'Haeseleer, P. "How does gene expression clustering work?" *Nature Biotechnology* 23.12 (2005), pp. 1499–1501.

[468] Gibbons, F. D. and Roth, F. P. "Judging the quality of gene expression-based clustering methods using gene annotation". *Genome Research* 12.10 (2002), pp. 1574–1581.

[469] Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., Walt, S. J. van der, Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R., Jones, E., Kern, R., Larson, E., Carey, C. J., et al. "SciPy 1.0: fundamental algorithms for scientific computing in Python". *Nature Methods* 17.3 (2020), pp. 261–272.

[470] Sokal, R. and Michener, C. "A statistical method for evaluating systematic relationships". *Univ Kans Sci Bull* 38 (1958), pp. 1409–1438.

[471] Mirisola, M. G., Braun, R. J., and Petranovic, D. "Approaches to study yeast cell aging and death". *FEMS Yeast Research* 14.1 (2014), pp. 109–118.

[472] Kwolek-Mirek, M. and Zadrag-Tecza, R. "Comparison of methods used for assessing the viability and vitality of yeast cells". *FEMS Yeast Research* 14.7 (2014), n/a–n/a.

[473] Alao, J. P., Weber, A. M., Shabro, A., and Sunnerhagen, P. "Suppression of sensitivity to drugs and antibiotics by high external cation concentrations in fission yeast". *PLoS ONE* 10.3 (2015), e0119297.

[474] Parker, K. R. and Borstel, R. C. von. "Antimutagenesis in yeast by sodium chloride, potassium chloride, and sodium saccharin." *Basic life sciences* 52 (1990), pp. 367–371.

[475] McInnes, L., Healy, J., and Melville, J. "UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction" (2018).

[476] D'Auria, F. D., Laino, L., Strippoli, V., Tecca, M., Salvatore, G., Battinelli, L., and Mazzanti, G. "In vitro activity of tea tree oil against Candida albicans mycelial conversion and other pathogenic fungi". *Journal of Chemotherapy* 13.4 (2001), pp. 377–383.

[477] Bernier, S. P. and Surette, M. G. "Concentration-dependent activity of antibiotics in natural environments". *Frontiers in Microbiology* 4.FEB (2013).

[478] Mazu, T. K., Bricker, B. A., Flores-Rozas, H., and Ablordeppey, S. Y. "The Mechanistic Targets of Antifungal Agents: An Overview." *Mini reviews in medicinal chemistry* 16.7 (2016), pp. 555–78.

[479] Bourgon, R., Gentleman, R., and Huber, W. "Independent filtering increases detection power for high-throughput experiments". *Proceedings of the National Academy of Sciences of the United States of America* 107.21 (2010), pp. 9546–9551.

[480] Vo, T. V., Das, J., Meyer, M. J., Cordero, N. A., Akturk, N., Wei, X., Fair, B. J., Degatano, A. G., Fragoza, R., Liu, L. G., Matsuyama, A., Trickey, M., Horibata, S., Grimson, A., Yamano, H., Yoshida, M., Roth, F. P., Pleiss, J. A., Xia, Y., and Yu, H. "A Proteome-wide Fission Yeast Interactome Reveals Network Evolution Principles from Yeasts to Human". *Cell* 164.1-2 (2016), pp. 310–323.

[481] Kellis, M., Birren, B. W., and Lander, E. S. "Proof and evolutionary analysis of ancient genome duplication in the yeast Saccharomyces cerevisiae". *Nature* 428.6983 (2004), pp. 617–624.

[482] Hakes, L., Pinney, J. W., Lovell, S. C., Oliver, S. G., and Robertson, D. L. "All duplicates are not equal: The difference between small-scale and genome duplication". *Genome Biology* 8.10 (2007), R209.

[483] Force, A., Lynch, M., Pickett, F. B., Amores, A., Yan, Y. L., and Postlethwait, J. "Preservation of duplicate genes by complementary, degenerative mutations". *Genetics* 151.4 (1999), pp. 1531–1545.

[484] Stoltzfus, A. "On the possibility of constructive neutral evolution". *Journal of Molecular Evolution* 49.2 (1999), pp. 169–181.

[485] Lehtinen, S., Bähler, J., and Orengo, C. "Co-Expression Network Models Suggest that Stress Increases Tolerance to Mutations." *Scientific reports* 5 (2015), p. 16726.

[486] Lehtinen, S., Marsellach, F. X., Codlin, S., Schmidt, A., Clement-Ziza, M., Beyer, A., Bahler, J., Orengo, C., and Pancaldi, V. "Stress induces remodelling of yeast interaction and co-expression networks". *Molecular Biosystems* 9.7 (2013), pp. 1697–1707.

[487] Cheng, J. and Baldi, P. "A machine learning information retrieval approach to protein fold recognition". *Bioinformatics* 22.12 (2006), pp. 1456–1463.

[488] Shah, A. R., Oehmen, C. S., and Webb-Robertson, B. J. "SVM-HUSTLE - An iterative semi-supervised machine learning approach for pairwise protein remote homology detection". *Bioinformatics* 24.6 (2008), pp. 783–790.

[489] Dlakić, M. "HHsvm: Fast and accurate classification of profile-profile matches identified by HHsearch". *Bioinformatics* 25.23 (2009), pp. 3071–3076.

[490] Lees, J. G., Das, S., and Orengo, C. A. "A Domain-Based Machine Learning Approach for Function Prediction using CATH FunFams". *Function SIG* (2017).

[491] Zaman, R., Chowdhury, S. Y., Rashid, M. A., Sharma, A., Dehzangi, A., and Shatabda, S. "HMMBinder: DNA-Binding Protein Prediction Using HMM Profile Based Features". *BioMed Research International* 2017 (2017).

[492] Sprinzak, E., Sattath, S., and Margalit, H. "How reliable are experimental protein-protein interaction data?" *Journal of Molecular Biology* 327.5 (2003), pp. 919–923.

[493] Silva, E. de, Thorne, T., Ingram, P., Agrafioti, I., Swire, J., Wiuf, C., and Stumpf, M. P. "The effects of incomplete protein interaction data on structural and evolutionary inferences". *BMC Biology* 4.1 (2006), pp. 1–13.

[494] Kuchaiev, O., Rašajski, M., Higham, D. J., and Pržulj, N. "Geometric de-noising of protein-protein interaction networks". *PLoS Computational Biology* 5.8 (2009), e1000454.

[495] Zaki, N., Efimov, D., and Berengueres, J. "Protein complex detection using interaction reliability assessment and weighted clustering coefficient". *BMC Bioinformatics* 14.1 (2013), p. 163.

[496] Piovesan, D., Giollo, M., Leonardi, E., Ferrari, C., and Tosatto, S. C. E. "INGA: protein function prediction combining interaction networks, domain assignments and sequence similarity." *Nucleic acids research* 43.W1 (2015), W134–40.

[497] Moya-García, A., Adeyelu, T., Kruger, F. A., Dawson, N. L., Lees, J. G., Overington, J. P., Orengo, C., and Ranea, J. A. "Structural and Functional View of Polypharmacology". *Scientific Reports* 7.1 (2017), pp. 1–14.

[498] You, R., Yao, S., Xiong, Y., Huang, X., Sun, F., Mamitsuka, H., and Zhu, S. "NetGO: improving large-scale protein function prediction with massive network information." *Nucleic acids research* 47.W1 (2019), W379–W387.

[499] You, R., Zhang, Z., Xiong, Y., Sun, F., Mamitsuka, H., and Zhu, S. "GOLabeler: improving sequence-based large-scale protein function prediction by learning to rank." *Bioinformatics (Oxford, England)* 34.14 (2018), pp. 2465–2473.

[500] Lees, J. G., Heriche, J. K., Morilla, I., Ranea, J. A., and Orengo, C. A. "Systematic computational prediction of protein interaction networks". *Physical Biology* 8.3 (2011), p. 035008.

[501] Bilder, R. M., Sabb, F. W., Cannon, T. D., London, E. D., Jentsch, J. D., Parker, D. S., Poldrack, R. A., Evans, C., and Freimer, N. B. "Phenomics: The systematic study of phenotypes on a genome-wide scale". *Neuroscience* 164.1 (2009), pp. 30–42.

[502] Gigerenzer, G. *Gut feelings : the intelligence of the unconscious*. Penguin Books, 2014.

[503] Gligorijevic, V., Renfrew, P. D., Kosciolek, T., Leman, J. K., Cho, K., Vatanen, T., Berenberg, D., Taylor, B., Fisk, I. M., Xavier, R. J., Knight, R., and Bonneau, R. "Structure-Based Function Prediction using Graph Convolutional Networks". *bioRxiv* (2019), p. 786236.

[504] Bhardwaj, N. and Lu, H. "Correlation between gene expression profiles and protein-protein interactions within and across genomes". *Bioinformatics* (2005).

[505] Peng, J. and Xu, J. "Low-homology protein threading". *Bioinformatics* (2010).

[506] Colin, P.-Y., Kintses, B., Gielen, F., Miton, C. M., Fischer, G., Mohamed, M. F., Hyvönen, M., Morgavi, D. P., Janssen, D. B., and Hollfelder, F. "Ultrahigh-throughput discovery of promiscuous enzymes by picodroplet functional metagenomics". *Nature Communications* 6.1 (2015), p. 10008.