Assessment of a Deep Learning Approach to the Classification of Insects using DNA Sequences

Student ID: 10707266

A dissertation submitted to The University of Manchester for the degree of Master of Data Science (Mathematics) in the Faculty of Humanities

Department of Social Sciences
United Kingdom
2021

Abstract

The classification of biological specimens is paramount in realising biodiversity changes over time and space, yet methods to solve such problem are still in their infancy. Statistical models have been shown to improve on the current standard of species classification, character based methods. In this work, a Naive Bayes model and a Convolutional Neural Network were fitted to a collection of genetic sequences from the *Insecta* class. Due to the low sample size of DNA sequences discovered in the data analysis stage and the over-parameterisation of the resultant structure of the CNN,data augmentation was required to aid generalisation of the two models. The most generalised and best performing model was the CNN, which achieved a test accuracy of 76.4% with 5-sampled species, 3 synthetic mutations and 5 changes per synthetic mutation.

Contents

Chapter 1

Introduction

1.1 Background

We know only little about biodiversity - it is estimated that about 86% of all species on Earth still await description [1]. Biodiversity loss occurs at an increasing rate, it is 500 times greater than what researchers would expect from natural causes. Further, the current rates of extinction are estimated to be 1000 times the likely background rate of extinction [2]. Insects (*Insecta*) are the richest and most diverse class of all animal groups. Some biologists estimate the number of species of insects to be 5 to 6 million [3], whilst others believe uncertainties in insect characteristics make a plausible range impossible to determine[4]. In recent decades, species and in particular, insect biomass and abundance have been declining dramatically.

Identifying organisms based on their genetic material is an important task in understanding the phylogenetic diversity in an environment. Estimating and tracking changes in abundance and diversity of insects at the species level through time and space is critical to understand the underlying drivers of change and to devise possible mitigation strategies [5] [6]. Novel techniques such as DNA barcoding, new databases, and crowd-sourcing, could greatly accelerate the rate of species discovery [4].

Biological organisms are grouped into taxa based on shared characteristics. These groups are given a taxonomic rank; groups of a given rank can be aggregated to form a more inclusive group of higher rank, thus creating a taxonomic hierarchy. In the 1750s, the Swedish botanist Carl Linnaeus founded the current system of taxonomy, known as Linnaean taxonomy, a ranked system for categorising organisms which included the binominal nomenclature for naming organisms [7]. Linnaeus categorised organisms into a hierarchy of kingdoms, classes, orders, families, genera, and species based on shared physical characteristics. The category phylum was added to the classification scheme later with advances in evolutionary biology. Today, taxonomists arrive at taxa depending on the available data, methods vary from qualitative comparisons such as behaviours and physical features, to genetic relatedness using DNA sequence analysis [8].

An important question is whether a realistic biological, populational or phylogenetic model for DNA sequence analysis is necessary. More specifically, purely statistical approaches may also be able to efficiently assign query sequences to species names.

In this dissertation, the classification problem will be approached using well-known machine learning methods, using the aligned genetic sequences to classify recorded species of the *Insecta* class. First, a Naive Bayes classifier was built to compare with current literature. Since there is a lack of deep learning methods applied to this problem in current state of the art, a convolutional neural network was constructed to assess its validity and performance. The effect of augmenting the sequences was explored to improve generalisation of both methods.

1.2 Related Work

In 2003, Herbert et al. proposed the idea of genetic barcodes, short and standardised regions of a specimens DNA sequence [9]. They established that the mitochondrial gene Cytochrome c oxidase I (COI) can serve as the core of a global bioidentification system for animals. They show that COI profiles can ordinarily assign newly observed taxa to the correct phylum or order, by creating a comprehensive COI profile species-level assignment of lepidopterans (butterflies) was possible with a 100% success rate. Two years later, the Barcode of Life Data System (BOLD) (`www.barcodinglife.org`) was launched providing a repository for DNA barcode records and a workbench that aids the management and analysis of barcode data [10].

The BOLD Identification System (IDS) is a standard for animal classification. The IDS for animals accepts at least 500 base pairs from the barcode region of COI and returns a species-level identification when one is possible [11]. BOLD uses the Basic Local Alignment Search Tool (BLAST) algorithm to identify single base indels in the query sequence before aligning the protein translation through profile to a Hidden Markov Model of the COI protein. This is followed by a linear search of the COI reference library to find the nearest neighbours. Then, a Neighbour-Joining tree using the Kimura 2-parameter (K2P) distance is reconstructed on both this set and the query sequence in order to assess the relationship between the query sequence and its neighbouring reference sequences. Closely related species can show deep sequence divergence [9], therefore IDS returns a species identification if the query sequence shows less than 1% divergence to the reference sequence.

Alternatives to the standard barcoding procedure, used in the BOLD Identification Engine have been explored in the past 15 years. Ross et al. assesses the reliability of several different methods of species identification (BLAST, genetic distance, and tree-based methods) that constitute the standard barcoding rubric [12]. Their study showed that there was generally no best-performing method for all evolutionary scenarios. In 2009, Austerlitz et al. compared these results with supervised statistical classification methods [13], which included k-nearest neighbour (kNN), classification & regression trees, random forest and kernel methods. Using simulated and real data sets, they concluded no single method was superior but the 1-NN was the most reliable. When tested on 424 real COI sequences representing 61 species of Amazonian butterflies (approximately 7 samples per species), the 1-NN has a success

rate of 90.4%. They report that the results for the simulated dataset depends heavily on the sample size, length of the sequence and the conditions for the simulated evolutionary divergence.

Continuing the discussion, in 2014 Weitschek et al. considered a further set of supervised machine learning methods to classify species with DNA Barcode sequences. They consider a Support Vector Machine (SVM), the rule-based RIPPER, a decision tree C4.5, and a Naive Bayes method and compare the results to ad-hoc and established DNA Barcode classification methods (the character-based BLOG [14], BLAST and neighbour joining). Out of the 8 empirical datasets used, 5 were obtained from BOLD containing multiple Barcode sequences obtained from the COI region for each species. Naive Bayes and SVM were the best classifiers with an average accuracy of 93% and 94% respectively; the Bats datasets performed the best (100%) with an average 10 samples per species. The details of the Naive Bayes classifier such as the prior distributions or marginal likelihood are not disclosed as the models were implemented using the software suite Weka. On synthetic data, simulated by considering time of species divergence and the effective population size, the Naive Bayes and SVM classifier outperform the traditional barcoding models. On empirical data, the results were comparable. Nevertheless, these results and other projects [15] establish the extensive validity of the application of supervised learning methods for species classification.

A naive bayes classifier, the Ribosomal Database Project Classifier, has also been used to classify bacterial 16S rRNA sequences in 2007 [16]. The sequences averaged 1,454 bases in length and were originally classified into 1,187 genera by the National Center for Biotechnology Information. Wang et al. treats the problem as a word-based classification scheme. They defined words as a set of bases lengths 6-9, calculating the word-specific prior as the ratio of the number of sequences containing the word with the total number of sequences. They tested the classifier with leave-one-out testing on continuous regions of 400, 200, 100 and 50 bases chosen at random from the test sequence. With bootstrap analysis, they obtained the best accuracy using the full-length sequence with 95.1% at the family-level and 91.4% and the genus-level, reporting 97.5% of taxon assignments matched in 95 or more of the 100 bootstrap trials, and these assignments were correct 98% of the time. Although classification at the species level was not discussed and the genetic sequence of interest is different, the methodology is notable as it takes from techniques in Natural Language Processing. There seems to be endless applications of machine learning and artificial intelligence to solve this biological classification problem.

In the studies described above, the classification is mainly performed within a specific taxonomic class assuming apriori knowledge about the given to-be-classified sequence. As the best classifier, SVMs, suffers from scalability issues, Sohsah et al. proposed a scalable, hierarchical approach to classify species in 3 different classes [17]. The first level uses an SVM to predict the correct class, then three further SVM predict the species given the previous class assignment. They obtain only 122 species for the *Chirptera* class, 127 for the *Rodentia* class and 841 for the *Aves* class, containing 4731, 3653 and 4192 sequences respectively. Similar to Wang et al, this hierarchical SVM was tested on sets of sub-sequences up to length 7. The test accuracy of the best performing model, where sub-sequences are 5 characters long, is approximately 93%. These are promising results, but their proclamation of a scalable

approach is questionable when 127 species are considered in a class containing more than 2000.

Deep learning methods have yet to reach the DNA sequence classification problem, however convolutional neural networks (CNNs) have been utilised for species-level classification of images of beetles [5]. The results indicate that habitus images are sufficient to classify images to species level, albeit not for all species due to body size variability. As well as biodiversity research, CNNs have been used to distinguish insects that cause natural disasters that affect crop yeilds [18].

It is clear that statistical approaches have their place in species classification, but they do possess a number of limitations. Since genetic sequences can be up to 900 bases long, they can be classed as high dimension data. At high dimensions, data is sparse so the training data may not capture all combinations of each feature. These difficulties arising from high-dimension data in machine learning is referred to as the *Curse of Dimensionality*. Weitschek at al. suggest that datasets containing at least 4 specimens per species must be used in order to achieve meaningful classification results. In deep learning, it is suggested that 1000s of training cases are required depending on the domain and quality of the data.

What can be learned from current literature is that classification becomes easier as you move up the genetic tree, results are strongly dependent on the biological group due to evolutionary divergence, and synthetic sequences tend to perform better than measured sequences. A generalised approach is required which considers an entire class and exploits the advantages of using synthesised data.

1.3 Report Structure

The remainder of this dissertation is split into 5 further chapters which presents the processing, results and discussions of the project. Chapter 2 breaks down the origin of the data utilised, issues with data quality and an exploratory data analysis. Chapter 3 explains the fundamentals of the Naive Bayes and Convolutional Neural Network models used for classifying the processed data. Due to the low sample size of DNA sequences discovered in the data analysis stage and the over-parameterisation of the resultant structure of the CNN, data augmentation was required to aid generalisation of the two models. The last section of Chapter 3 describes the methods of augmentation applied to the models. In Chapter 4, the simulations are described and the results are presented, summarising the performance of the models by the test accuracy and f1 score. Following the results, the next chapter dissects the observations and provides some suggestions for further work, given the effectiveness of augmentation. Finally, the last chapter concludes the dissertation.

Chapter 2

Data

2.1 Origin

Lifeplan are a consortium based at the University of Helsinki set up to establish the current state of biodiversity across the globe and to generate accurate predictions of its future state under future scenarios `https://www2.helsinki.fi/en/projects/lifeplan`. They generate a globally distributed data set on a broad range of taxonomical groups with the help of collaborators around the world. Additionally, they employ modern sampling methods that do not require taxonomical expertise from those collecting the data, which results in data that are directly comparable among different locations. This project was undertaken in collaboration with Lifeplan. The data used in this work was a copy of the BOLD library from 2018, provided by Lifeplan.

The raw data contain DNA barcodes sequences obtained from the Barcode of Life Data System, a freely available and collaborative database and workbench that supports the assembly and use of DNA barcode data, introduced above. Users can submit specimen records such as sequences, images and trace files pending acceptance from the BOLD data management team. The protocol for submission states a minimum requirement for specimen data which includes a sample ID, storing institution, phylum and country [10].

The Barcode Index Number System is an online framework that clusters barcode sequences algorithmically, generating a web page for each cluster. These clusters are indexed with a unique indentifier, the Barcode Index Number (BIN) e.g. BOLD:ADI1757. Since clusters show high concordance with species, this system can be used to verify species identifications as well as document diversity when taxonomic information is lacking. BINs refer uniquely to DNA sequences, but taxonomic classification does not. Multiple species can be found in a BIN and species can be observed in multiple BINS.

Professor David Spiegelhalter OBE FRS, a British statistician devised a star rating to assigned to the quality of data [19]. The ordinal scale starts at 0-stars which represents data that have been made up, such as urban legends or fabricated experimental data. A 4-star rating, the highest score, would be given to the most accurate type of data sets such as

official statistics. Whilst the sequences here are measured in controlled laboratory experiments, there is some possibility of measurements error yielding the cases where species names are uncertain (more below). Additionally, annotations may differ across the institutions/labs that upload the sequences, where BINs may refer to different species or even different orders. This data could be classified as a 3-star rating.

2.2 Preprocessing

There were two source files, the first contains a text file of genetic sequences with their corresponding BIN. First, the genetic information with well-defined bases ($AGCT$) and sequences of length 901 were extracted. Other, ambiguous nucleotides such as R, indicating Purine (A or G) [20] were not considered for accuracy and memory purposes. Subsequently, there were 520392 sequences remaining. The second file contained a dataframe containing the taxonomy of species also indexed by a BIN, with columns: *Phylum, Class, Order, Family, Genus, Species*. This data was reduced to only contain the BINs present in the sequences file.

The genetic sequences considered in this dissertation is a string of the characters $AGCT$ and a dash indicating a gap in the DNA sequence. A excerpt of a typical sequence is

> ————————-ATTATATATTTTATGTTTGGTTTGTGATCTGGAATACTAGGAT
> TTTCAATG—AGTTTAATTATTCGTTTAGAATTAGGTAATCCAGGAAGATT
> AATTGGTAAT ...

As biological sequences evolve, sequences with the same ancestral history may have differing lengths due to replaced, added or deleted bases. To ensure that bases with the same evolutionary history are found in the same column, the sequences are aligned using Hidden Markov models that result in the inserted gaps [21].

2.3 Data Quality

Open nomenclature in biodiveristy states that where a species has not been identified down to the species level, the genus name should be followed by *sp.*. If there are distinct unidentified species in a genus then they should be labelled further by a figure or character e.g. *Bleptina sp. 1.* The abbreviation *cf.* followed by the species name indicates that most of the diagnostic characters correspond to a given species, but some characters are unclear, therefore identification is provisional [22]. These codes are present in the data but there are many instances where species names are given that do not follow standard nomenclature. Table 2.1 show the frequency of each type of naming inconsistency.

To maintain clarity in this classification problem, the species name containing *sp.* were relabelled to N/A and removed from further analysis. As this list of irregularities is not exhaustive, it would be a substantial task to remove each instance manually. Therefore the sequences were filtered such that only the species with more than 1 sample present were kept.

Naming inconsistency	Count
Contains cf.	1060
Spelling Error	Unable to determine
Contains full BIN	2325
Contains partial BIN	5287
Contains inconsistent punctuation characters	Unable to determine
Contains n. or nr.	14003

Table 2.1: Inconsistencies in the naming of species discovered at the pre-processing stage.

2.4 Exploratory Data Analysis

The insects data set, the sequences representing species from the *Insecta* class, contains 28 Orders, 963 families, 26596 genera and 171621 species. After removing the ambiguous species names, the number of unique species reduces to 168529. This number decreases rapidly as the species with multiple sequences are considered, see Figure 2.1. Approximately 0.13% of species in the data set contain 11 or more genetic sequences and 1.4% contain 4 or more as recommended by Weitschek et al (2013). With increasing sample size the issue of generalisation needs to be addressed. When the sequences themselves are analysed,

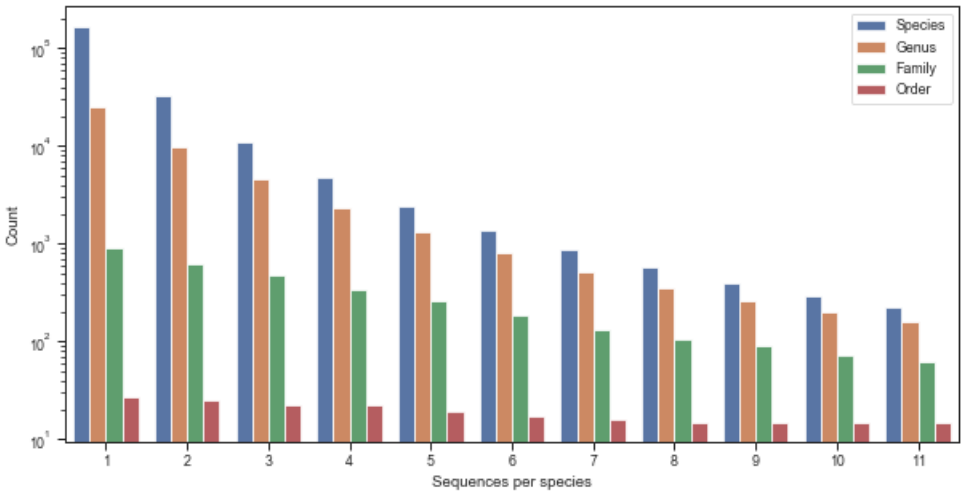

Figure 2.1: Number of unique species in the dataset remaining as the sample size of each species considered increases.

the most common nucleotide is typically T, while the least common being G, depending on the Order of interest. Table 2.2 gives a total breakdown of sequences by Order. At a glance, many Orders have very similar configurations, for example, the occurrence of bases in *Hymenoptera* (i.e. wasps, bees & ants) and *Trichoptera* (i.e aquatic larvae) sequences differ by no more than 12. The outliers to these arrangements are not due to vastly different characteristic of the insect, but as a result of the number of gap characters. The *Mantodea*

group (mantids) contain 247 gap characters in their sequences, while one of their closest relatives the *Blattodea* group (cockroaches) has only 224 gap characters. It is clear that the gaps are vital in distinguishing specimens, but inconsistencies further down the phylogentic tree may provide difficulties in classifying closely related species.

Order	A	T	G	C	-
Lepidoptera	199.51	256.71	97.06	103.36	244.36
Depressariidae	197.03	254.80	96.76	102.52	249.89
Gelechiidae	195.12	256.49	96.57	101.68	251.14
Pterophoridae	199.21	250.62	93.68	104.19	253.30
Hymenoptera	206.33	261.04	81.34	103.88	248.41
Trichoptera	198.40	247.01	90.51	116.28	248.80
Psocodea	177.86	264.70	129.05	121.28	208.11
Neuroptera	185.46	251.58	106.00	112.54	245.42
Blattodea	210.68	195.57	117.75	153.03	223.96
Mantodea	204.42	233.85	98.76	117.05	246.91

Order	A	T	G	C	-
Lepidoptera	14.58	17.12	7.21	9.98	36.95
Depressariidae	8.36	9.48	4.23	8.94	14.92
Gelechiidae	8.44	10.30	4.32	7.54	17.56
Pterophoridae	10.68	11.93	5.26	8.62	23.74
Hymenoptera	20.87	24.57	12.06	23.20	42.91
Trichoptera	12.77	16.31	9.50	16.75	20.71
Psocodea	23.43	44.14	36.03	19.69	94.10
Neuroptera	18.80	24.82	9.07	14.33	48.77
Blattodea	25.62	30.57	14.90	24.44	70.70
Mantodea	16.09	30.90	13.05	11.20	62.69

Table 2.2: Left: Average occurrences of each nucleotide and gaps present in the genetic sequence grouped by Order in the Class *Insecta*. The Order Lepidoptera is broken down by representative Families. Right: Standard deviations for the same taxon. The lists of orders and families are not exhaustive and only represent a subset of all taxon in the Insecta data set.

2.4.1 Similarity Investigation

The python libraries TreeMaker [23] and BioPython [24], were employed to visualise the phylogenetic tree of the insect dataset, see Figure 2.2. A bug in the BioPython package meant that branches with a single species at the last node would cut short e.g. the *Gelechildae* branch would stop at the family level. Therefore, a function was written to edit the newick files, wherever a single species or genera is plotted, so that the entire taxonomy is shown.

The Levenshtein distance is a metric assessing the similarity between two strings, calculated as the count of the minimum number of substitutions and single letter insertions or deletions required to convert one string to another [25]. The metric is used in biological sequence analysis as a proxy for the evolutionary distance between two sequences [26]. To understand the similarities between sequences of species close to each other in the phylogenetic tree, the Levenshtein distance was calculated between the species in the Insecta class 2.3.

As the species in Figures 2.2 and 2.3 are in order, one would expect species next to each other to be similar due to their taxonomic placement. Along the diagonal, the darkest patches come from *Nylanderia amblyops* & *Nylanderia bourbonica*, both belonging to the *Nylanderia* genus and *Melese Chozeba* & *Lopocampa districta*, both of the *Erebidae* family. The majority of sequences differ by 100 characters or less - all of which are in the *Lepidoptera* order. The lighter bands to the left and top of the plot indicate difference of at least 150 characters, highlighting the deviation of sequence characteristics at the order level.

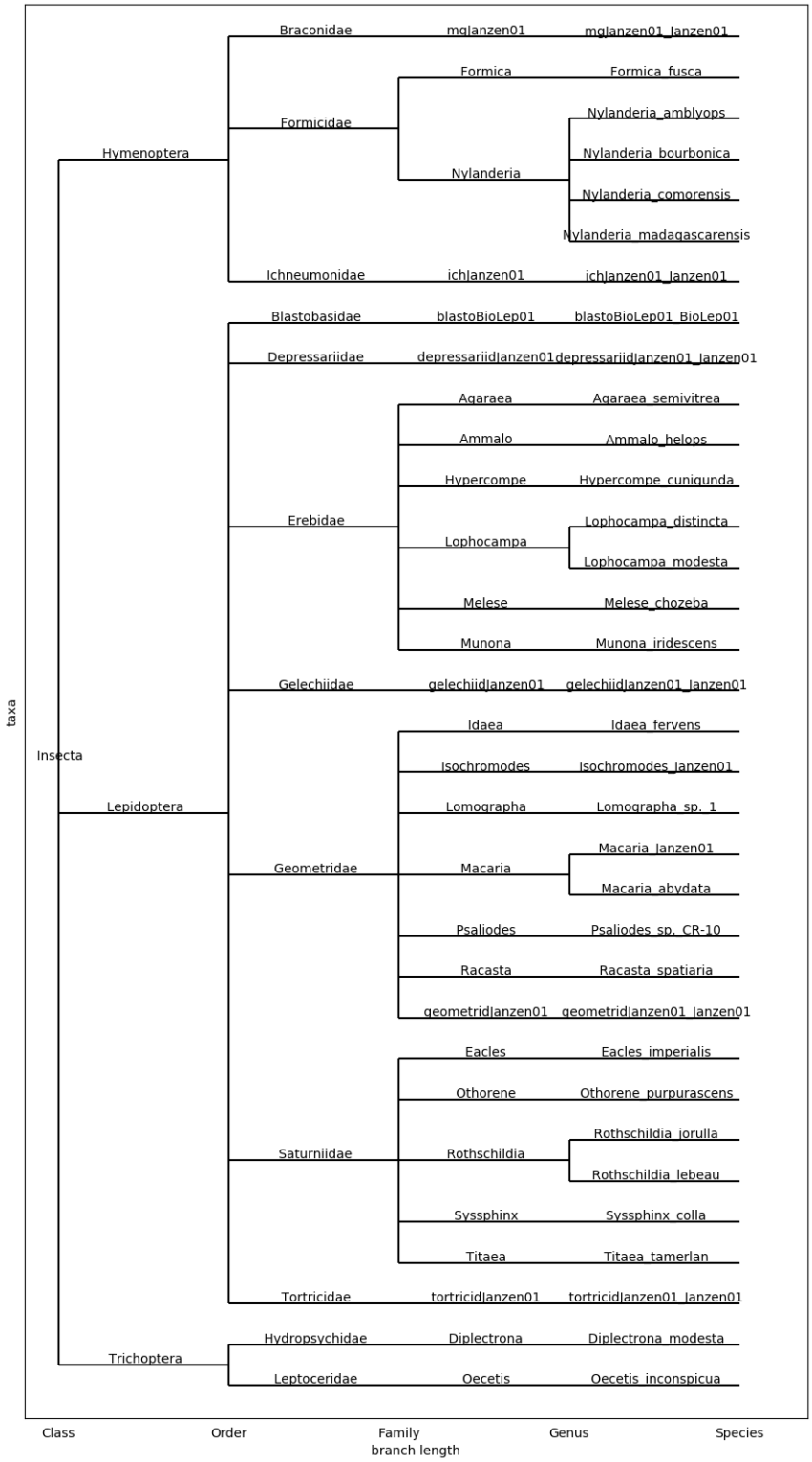

Figure 2.2: Representative phylogenetic tree from the Insecta class. Only a subset of 40 species were considered; modes will typically have more leafs at the species and genus level.

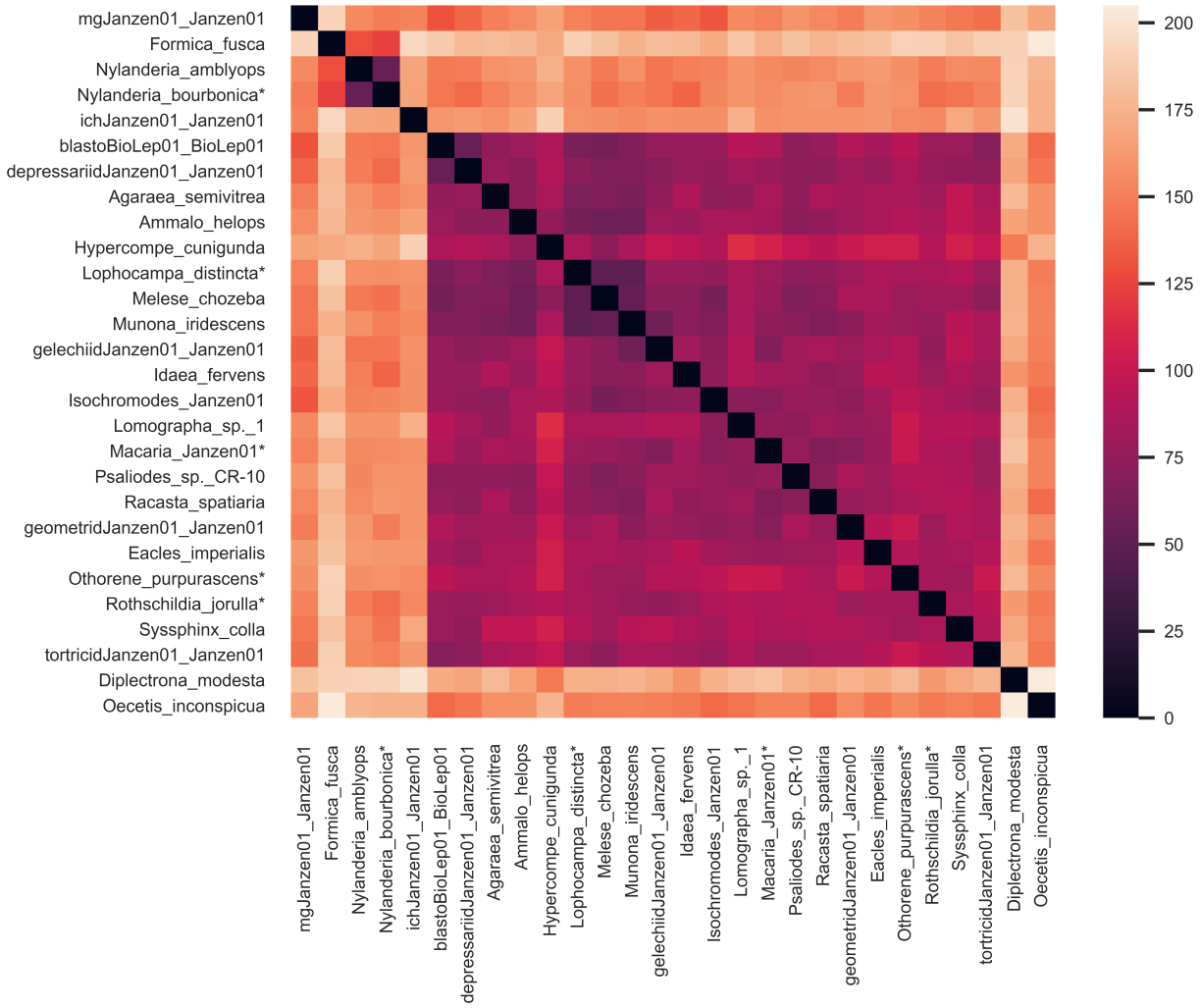

Figure 2.3: Similarity between the species of the Insecta class presented in the genetic tree in the same order, calculated using the Levenshtein distance. The species/genera with an asterisk denote the set of differently labelled species with the same genetic sequence.

Chapter 3

Methodology

3.1 Naive Bayes

One of the most effective classifiers is the so-called Naive Bayes classifier described by Duda and Hart [27] and Langley et al. [28]. Classification is done by applying Bayes' rule to compute the probability of a class given the particular attributes and then predicting the class with the highest posterior probability.

The task is to predict taxon in a taxonomic rank, C, for an unseen sequence x_{new}. From Bayes' rule, one can obtain the expression for the Bayesian classifier which gives the probability of sequence belonging to taxon $c \in C$, given training sequences $X = [x_1, x_2, ..., x_n]$ with associated taxon $t = [t_1, t_2, ..., t_n] \in C$:

$$p(c|x_{new}, X, t) = \frac{p(x_{new}|c, X, t)p(c|X, t)}{\sum_{c' \in C} p(x_{new}|c', X, t)p(c'|X, t)} \tag{3.1}$$

The prior distributions, $p(c|X, t)$ were estimated by the number of sequences belonging to taxon c over the number of sequences in the training data, n. Although this is an estimation, it will be close to the true prior probability by the strong law of large numbers.

The naive assumption that base occurrence in a sequence are independent was utilised, henceforth the taxon conditional distribution can be be factorised into a product of univariate distributions [29]. There are 901 bases in a sequence, but after one-hot encoding there are M=901x5 features of 0s and 1s:

$$p(x_{new}|c, X, t) = \prod_{m=1}^{M} p(x_{new,m}|c, X, t) \tag{3.2}$$

where $x_{new,m}$ denotes the value in the test encoded sequence at position m. Using a multivariate Bernoulli likelihood, for $b \in [0, 1]$:

$$p(x_{new,m}|c, X, t) = p(b|c, X, t)^{x_{new,m}}[1 - p(b|c, X, t)]^{1-x_{new,m}} \tag{3.3}$$

where the following Laplacian prior is calculated at the training stage.

$$p(b|c, X, t) = \frac{1 + M_{b,c,m}}{2 + M_c} \tag{3.4}$$

where $M_{b,c,m}$ is the number of training sequences in taxon c that contain a base flag $b(= 1)$ at position m, and M_c is the total number of training sequences in taxon c [30].

3.2 Convolutional Neural Network

Convolutional Neural Networks (CNN) are a regularised multilayer perceptron (MLP) that and can capture the spatial and temporal dependencies in an image, introduced by Yann LeCun in the 1990s [31]. Typically the input, a 2-D matrix for grey-scaled images, are condensed by applying the dot product with kernel matrices iteratively through the matrix space creating the convolution layer [32]. The size of the convolution depends on the kernel size, number of kernels, the stride and padding of the input image. This 3-D matrix is transformed using a non-linear activation function and further reduced by pooling – where a subset of the matrix (size determined by the pooling size hyperparameter) is typically summarised by an average or the maximum value. The convolution process is repeated with another set of kernels, flattened and reduced with several MLPs to obtain the posterior probability distribution over the output classes. The Softmax activation function returns the most probable choice for multi-class problems. The network uses gradient descent applied to the sum-of-squares error function to adjust the weights with each state denoted as an epoch.

The input image in this task is a one-hot encoded version of the sequence data, with the gaps expressed as [0,0,0,0]. To obtain an even dimension for convolution, the first character in the sequence was removed, resulting in a shape of (900, 4). The first convolution uses 20 kernels of size 2x2 which explores the space in strides of 2. Since the input image is long and thin, no padding was added which may suppress the effect of the true data. This is followed by a maximum pooling layer, with pool size 2x2 and stride 2, and a ReLu activation layer. The second convolution layer was identical, now with 50 kernels. Next, the data is flattened and condensed in 1 hidden layer using a ReLu activation layer. The dense layer is a neural network layer that is connected deeply, which means each neuron in the dense layer receives input from all neurons of its previous layer. Lastly, a softmax activation function returns the species with highest probability.

An adaptive learning rate algorithm was chosen as the optimiser in the gradient descent stage. They achieve better results for sparse data inputs [33] compared to stochastic gradient descents which are slow and have a risk of converging to local minimums. In addition, there is no need to tune the learning rate. Adaptive Moment Estimation (Adam) was selected with a learning rate of 0.001.

The number of hidden cells directly impact the accuracy of the network, too few nodes will lead to a high error in the system if there are a many predictive factors and too many nodes will lead to overfitting at training so the network will not generalise well. The rule of thumb

for choosing the number of hidden nodes, is [34]

$$(\text{input layer size} + \text{output layer size}) \times 2/3 \tag{3.5}$$

The input layer size is always 900 but the output layer size is dependent on the number of sample per species considered.

Randomly removing nodes at the training stage can reduce overfitting and hence the test loss. A dropout layer, where the nodes are removed with a given probability, was included after both convolutional layers. Brownlee et al. states that for input units, the optimal probability of retention is usually closer to 1 than to 0.5 [35]. To confirm this, the CNN was tuned to find the combination of 2 dropout probabilities which maximised test accuracy. This yielded the first dropout probability of 0.3 and the second of 0.1.

3.3 Data Augmentation

Augmenting data is a widely used technique to artificially expand the sample size, particularly in object recognition tasks where augmentation may be in the form of rotations, dimension reordering, ZCA whitening and random shifts. The aim is to feed additional, plausible variations of the training set to the model. The result is a more generalised fit. Image data augmentation is now supported in the Keras deep learning library via the ImageDataGenerator class `https://keras.io/api/preprocessing/image/`.

The sequence data are not as complex as image data when they are encoded into tensors, therefore the augmentation techniques are limited. To emulate a mutation in the sequence, a position is chosen at random and the base is altered. The input data for the following two methods were slightly different which required slight different synthetic mutations. For the Naive Bayes model, the encoded data is a 1-D array, hence a position the array is chosen at random and the value at that position is switched from 0 to 1 or vice versa. The input sequences to the CNN are 900 tensors of size 4, as explained in Section 3.2. After a tensor is selected at random, the existing tensor is replaced by a different one also at random e.g. [0,0,0,1] may change to [0,1,0,0]. These synthetic mutated sequences are exclusive to the training of both models i.e. no synthetic mutations were applied to the test sequences.

Layer (type)	Output Shape	Param #
conv2d_8 (Conv2D)	(None, 450, 2, 20)	100
activation_16 (Activation)	(None, 450, 2, 20)	0
max_pooling2d_8 (MaxPooling2	(None, 225, 1, 20)	0
dropout_8 (Dropout)	(None, 225, 1, 20)	0
conv2d_9 (Conv2D)	(None, 225, 1, 50)	4050
activation_17 (Activation)	(None, 225, 1, 50)	0
max_pooling2d_9 (MaxPooling2	(None, 113, 1, 50)	0
dropout_9 (Dropout)	(None, 113, 1, 50)	0
flatten_4 (Flatten)	(None, 5650)	0
dense_8 (Dense)	(None, 602)	3401902
activation_18 (Activation)	(None, 602)	0
dense_9 (Dense)	(None, 4)	2412
activation_19 (Activation)	(None, 4)	0

Total params: 3,408,464
Trainable params: 3,408,464
Non-trainable params: 0

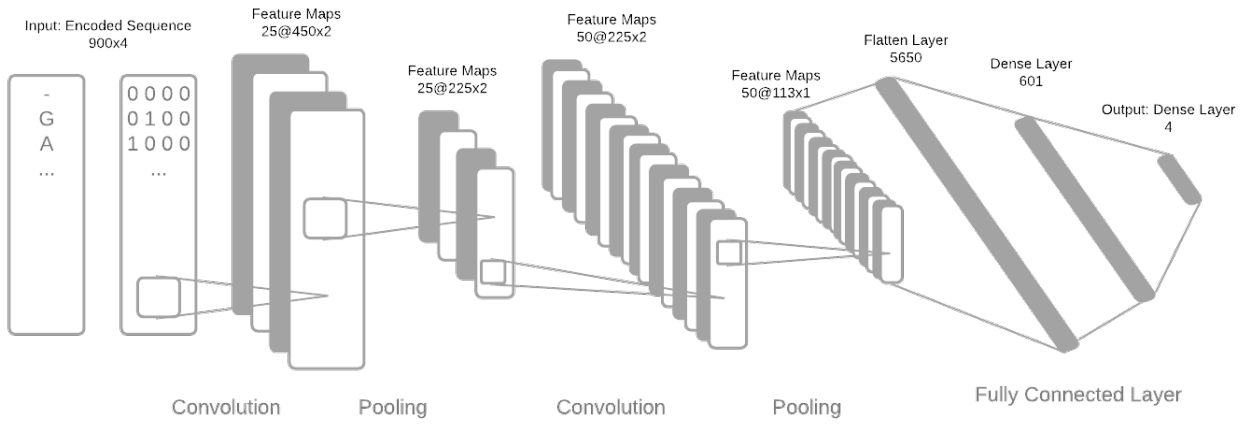

Figure 3.1: Example structure of the CNN built for input sequences of size 900x4, where species with at least 40 sequences are considered. The first convolution explores the 900x4 latent space with 20 kernels size 2x2 with a stride of 2. The number of parameters are calculated by adding the weight parameters with biases: (input channels $\times$ output channels $\times$ kernel height $\times$ kernel width)+output channels, which yields (1x20x2x2)+20=100. The same calculation can be done to obtain the parameters for the second convolution. At the dense layer, vector-vector multiplication is performed, so the number of parameters that are trained equate to the length of the input vector multiplied by the length of the output vector plus the biases.

Chapter 4

Experimentation

4.1 Environment

All simulations is this section were carried out through HTCondor, the University of Manchester's high throughput computing service. The Naive Bayes was perform using the *BernoulliNB* method within scikit-learn version 0.24.2. The CNN was built using keras 2.3.0 supported by tensor flow version 2.0.

4.2 Naive Bayes

Naive Bayes classification was performed on the insect data set filtered to contain species that have at least 3 sample sequences at the family, genus and species level. This was repeated with up to 4 synthetic mutations appended to the training sequences. Prior to model fitting, the data was split into train and test data sets with a ratio 70:30. The test accuracy of classification is given in Figure 4.1. Some points are missing due to the memory limits when creating large Numpy arrays.

The f1 score for classification at the species level is very poor when sample size is at least 3, increasing as the sample size increases. Acceleration of model accuracy decreases as you move up the genetic tree, until the family level which stagnates at 81%. Using the information from Figure 2.1, the subset of data with sample size of at least 11 represent 7% of available families, 1% of available genera and 0.1% of available species. While the performance increase, the model becomes less generalised to the *Insecta* class.

Adding synthetic mutations increases the performance of the model at all taxonomic levels, with the species level seeing the most impact. Improvements diminish at 4 mutations, which is consistent across all taxonomic level. If the mutations were too significant such that they cause an overlap in the sequences for closely related species, the family and genus level would not diminish as quickly as the species level. This indicates there is a limit of model performance using single-change mutations, which is reached at around 4 or 5 mutations.

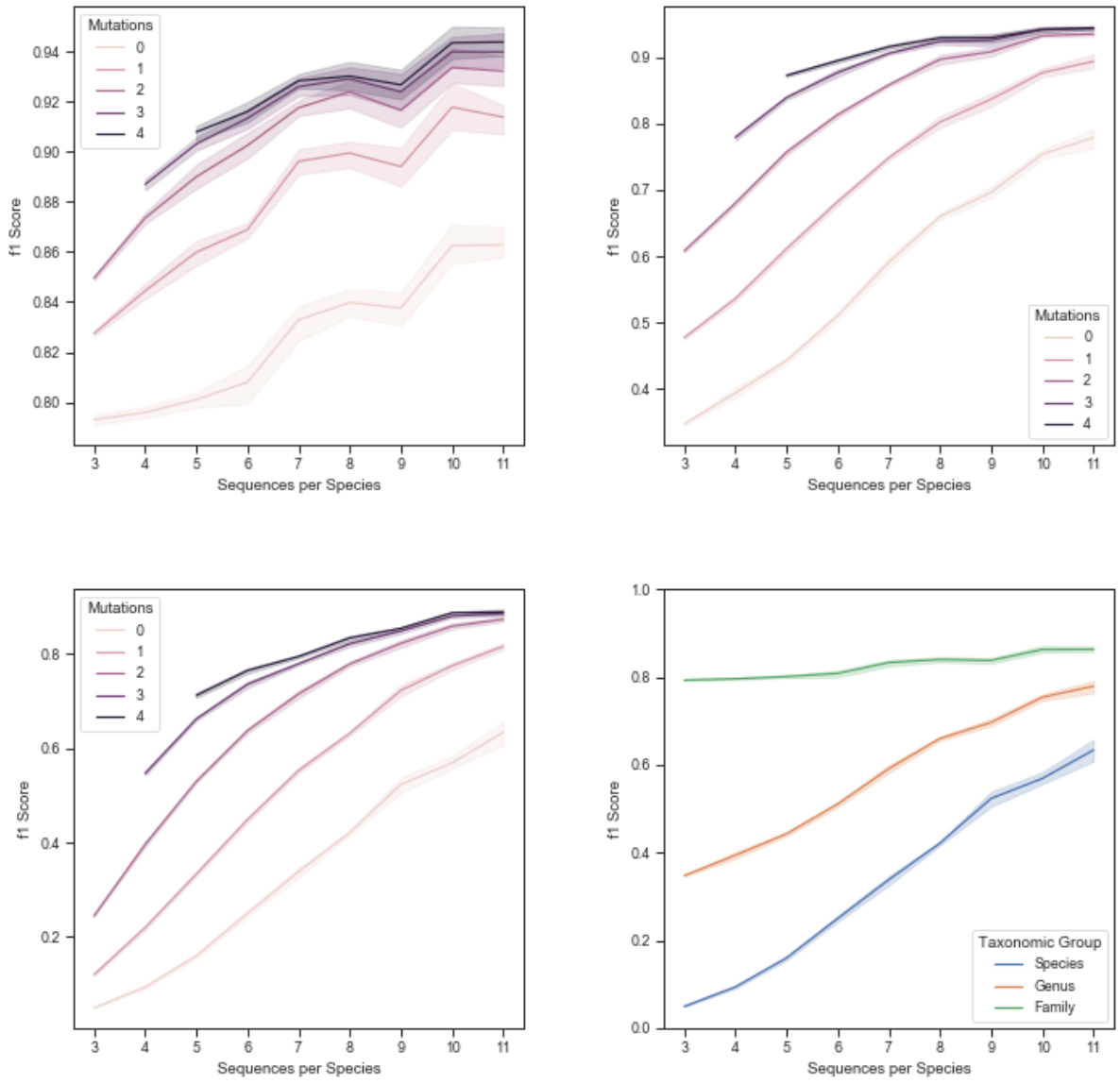

Figure 4.1: Average f1 score of test classification for the Naive Bayes model at the Family (top-left), Genus (top-right) and Species level (bottom-left). Bottom right: Average f1 score using training data with 0 mutations. The f1 score was calculated as a weighted average of the precision and recall, the classification was repeated using 3 random seeds at the train/test split stage.

Since the accuracy level at high taxonomic groups were observed, a sequential Naive Bayes model was attempted which attempted to classify species by cascading down the genetic tree. First the model was applied to the order level (which obtained an f1 score of 93% using sequences with no mutations) to obtain a predicted family. A new set of family-level Naive Bayes models were trained with the same training subset as before, the classification of the order-level determines the next model the sequence gets tested on. This goes to until classification can occur at the species level. There were no improvements to the results.

4.3 CNN

Once the CNN was built, it was trained on two subsets of the insect data: species attributed to at least 5 sequences and 10 sequences, including synthetic mutations, to discover to what effect augmentation has on test accuracy. To preserve relative species frequency in the train and test sequences, stratisfied sampling was implemented.

Regarding the 5-sample species data, the test accuracy of the mutated data converges much faster than the original sequences to a final accuracy 2% points higher. Further, the gap between train and test accuracy increases with mutations indicating the model is overfitting more. With the corresponding loss plot, the loss for the mutated sequence data converges much faster than the original sequences, as before, but then the loss steadily increases with the number of epochs.

Similar patterns are observed in the 10-sample species data: a faster converging accuracy and loss score and slight improvements using mutations. Now with 6 mutations, the test accuracy does not improve on the 3 mutations results and the loss is strictly increasing. At this stage, ultimate overfitting is observed with the training accuracy reaching 1 after 3 epochs; no further mutations will benefit the model accuracy.

Changes per Synthetic Mutation	Test Loss	Test Accuracy
1	2.028	0.733
5	**2.004**	**0.764**
10	2.123	0.764
15	2.486	0.685
20	2.737	0.681

Table 4.1: Test accuracy and loss for 5 CNNs trained on the species in the insect dataset that contain at least 5 samples with 3 additional synthetic sequences per sample, measured at 20 epochs.

The inclusions of synthetic mutations are more impactful in the Bernoulli Naive Bayes classifier than the CNN. One reason for this observation may be due to the convolutions and pooling layers in the CNN which may eliminate the effect of a single switch in the encoded data. A greater number of changes per synthetic mutation may be required to counter this phenomenon. To investigate the extent to which mutations are getting lost in pooling, another CNN was fit with training data containing 3 additional synthetic sequences. This time with multiple changes per synthetic mutations. Table 4.1 shows that the test accuracy is

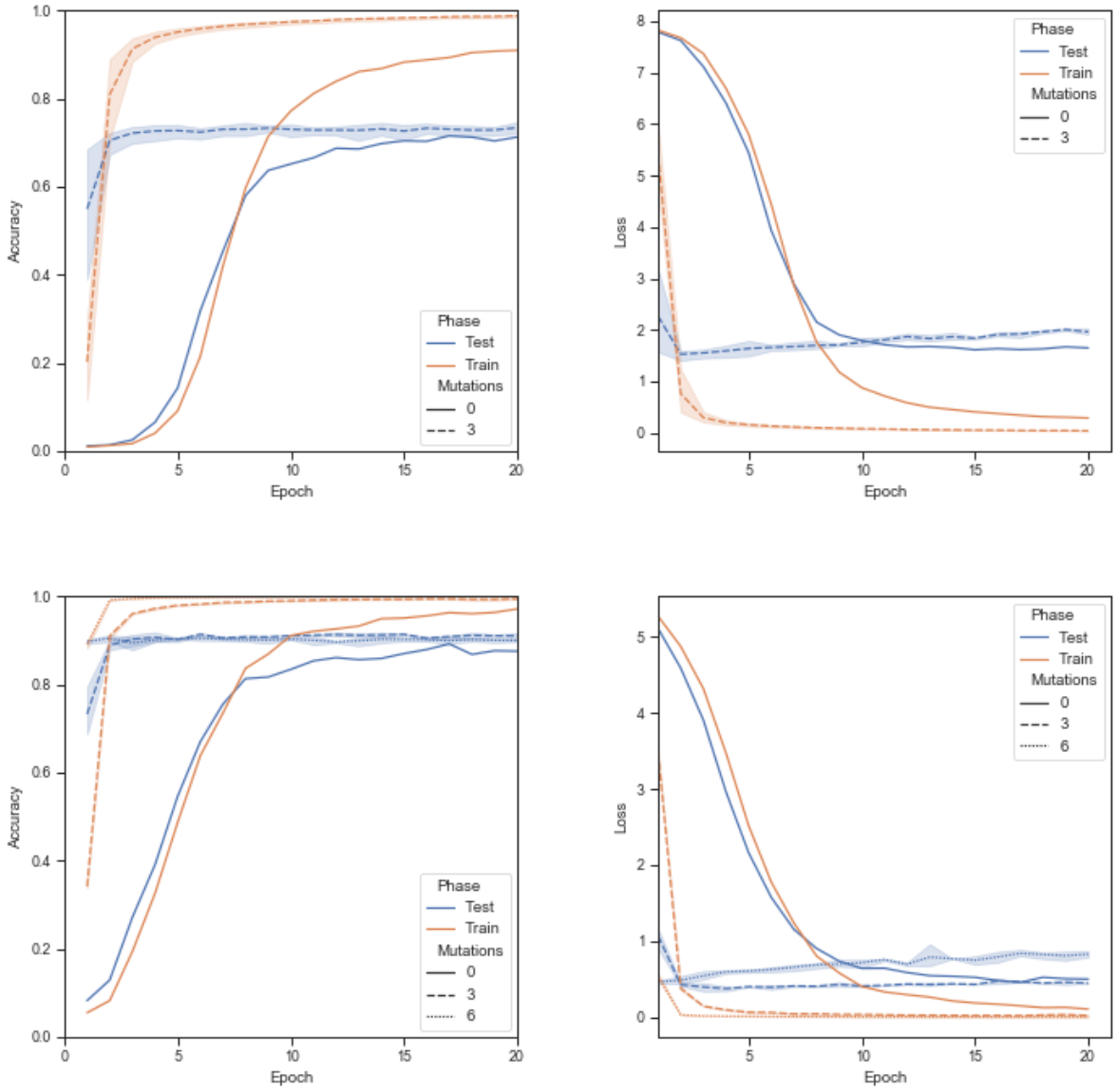

Figure 4.2: Converging accuracy and loss of the CNN as it undergoes backpropagation. Top: Subset of insect data consisting of the species attributed to at least 5 sequences. The dashed line show the results when the training dataset include 3 additional mutated versions of the original training sequences. Bottom: Subset of Insect data consisting of the species attributed to at least 10 sequences. As memory persisted with this smaller subset, training sequence with 6 mutations were simulated.

maximised between 5 and 10 changes per synthetic mutation, the test loss results suggest this value is closer to 5. As the synthetic mutations become more distinct, the accuracy decreases to a level lower than that of no mutations. At the point of 20 changes per synthetic mutation, the contrast between species will begin to blur as discovered in the similarity investigation. Nevertheless, this swift exercise has uncovered the highest performing and most generalised model.

Chapter 5

Analysis

From the results, the CNN performed much better than the Naive Bayes method. With the 5-sampled data and 3 single synthetic mutations, the Naive Bayes classification achieved 0.65 f1 score while the CNN achieved an accuracy of 0.73. It is likely that the Naive Bayes underperformed due to the conditional independent assumptions not holding in genetic sequences with 901 features.

The Wilcoxon signed-rank test is a non-parametric statistical hypothesis test used to compare two repeated measurements on a single sample to assess whether their population mean ranks differ [36]. In particular, the test can be applied to the test scores of two classifiers to determine if the difference between scores follow a symmetric distribution around 0 [37]. Thus, the following hypothesis can be deduced:

H_0: Both classifiers perform equally well.
H_1: Reject H_0

Let R^+ be the sum of ranks for the data sets on which the second algorithm outperformed the first, and R^- the opposite. Define T as the minimum of these two measurements, and N the number of species being classified. Then the test statistic

$$z = \frac{T - \frac{1}{4}N(N+1)}{\frac{1}{24}N(N+1)(2N+1)} \tag{5.1}$$

is normally distributed. To perform a two-sided test, the null hypothesis can be rejected at the 5% significant level if $|z| > 1.96$. First, comparing the f1 scores from the CNN with 0 mutations and the CNN with 3 mutations (top-left plot in Figure 4.2), a Wilcoxon statistic of 266831.5 was calculated (p-value $= 3.02 \times 10^{-7}$) so there is significant evidence at the 5% level to reject H_0. The CNN trained with 3 mutations has a greater accuracy, therefore it can be concluded that this model performs significantly better than the CNN trained with 0 mutations. Further, the Wilcoxon statistic between the CNN with 1 change per synthetic mutation and 5 changes per synthetic mutation was 209283.5 (p-value=0.00816). As the latter model obtained a greater test accuracy (Table 4.1), the null hypothesis can be rejected and the model with 5 changes per mutation performs significantly better than a single change.

Using the best performing and most generalised model in this project, see Table 4.1, 426 out of the 2460 species considered have a classification specificity of less than 0.25 and 177 have a precision of less than 0.25. The majority of species that are classified with poor specificity and precision come from the *Formicidae*, *Geometridae* and *Saturniidae* families. Likewise, the genera with poor specificity and precision are *Pheidole*, *Strumigenys* and *Automeris*. These taxon also happen to be the most populated families and genera in the insect data set. This observation may be due to the genera containing more species, so there are more instances where a species has a very low sample size (at least 5 + 15 including the synthetic mutations) and the model cannot generalise well for these species. Alternatively, there may be so much inter-genus diversity that the species are too similar or overlap in characteristics, particularly when the sequences are augmented.

Generally, misclassification in this model may be due to an error in the underlying taxonomy from the BOLD System. As discussed before, these sequences are labelled with a Barcode Index number which refer uniquely to taxonomy. In this dataset species are attributed to multiple BINs and, in some instances, BINs represent more than one species. To avoid these inconsistencies, the same exercise could be performed on COI sequences before they have been aligned by BOLD's classifier, as performed in previous studies [17] [15] [38]. The other main source of error in this project has been the sample size of the species available in this dataset. To gain further insight, an up-to-date data should be used. The BOLD system is continuously added to by biologists all over the word, to date the public data portal contains 5,883,735 sequences, representing 204,524 species. One could use the BOLD's API to obtain the COI sequences for current species. Users can query the system to retrieve matching sequence data records for a combination of parameters such as taxon, geographic location and institution.

While previous studies test their prospective machine learning methods on empirical and synthetic data independently, in this work data augmentation allowed the extension of training of real sequences. Two factors were explored: number of synthetic mutations and the number of changes per mutation, both improving the performance of the CNN. However, the results do show that the risks of overfitting increases with the number of synthetic mutations, indicating that the mutations do not add enough variation to the training sequences. Other, more drastic augmentation techniques are likely to improve generalisation even further:

- Shifting: By moving the 900x4 encoded data, the CNN will have more chances to learn the patterns in the sequence for given species. However, this variation will not be present in the testing data if the sequences are aligned. COI sequences come in a range of lengths so shifting will likely enhance classification performance of these sequences.
- Informed Mutations: At the synthetic mutation stage, the position of the sequence and the new nucleotide are chosen at random. In many cases a location that universally held a gap character would have been chosen, creating little variation to the training sequence. By using a prior as in Equation 3.4, one could obtain the most likely positions where variations are more common, akin to Monte Carlo sampling.
- Distortions: Another interesting investigation would be to distort the 1s and 0s in the encoded sequences by adding white noise with varying standard deviations.

Other techniques such as reflecting may be too much of a change. The results state that mutations with 20 changes decrease the predictive power of the models, suggesting this is the average Levenshtein distance between closely related species. To automate the augmentation stage, Generative adversarial networks (GANs) could be applied, which generate more examples from an estimated probability distribution from training examples [39].

A worthwhile task for future work would be to reduce the data to ease model complexity. Dimensionality reduction yields a more compact, more easily interpretable representation of the target concept, focusing the user's attention on the most relevant variables. In addition, less degrees of freedom will reduce the likelihood of the models overfitting to the training data. Uniform Manifold Approximation and Projection (UMAP) is a dimension reduction technique that can be used for visualisation similarly to t-SNE, but also for general non-linear dimension reduction [40]. UMAP is among the fastest manifold learning implementations available-significantly faster than most t-SNE implementations. Manifold learning methods seek a lower-dimensional projection of high dimensional input that captures the salient properties of the input data. It is built based on the assumption that the data is uniformly distributed, the Riemannian metric is locally constant and the manifold is locally connected. UMAP can be applied to this data using the *umap-learn* package in Python. Alternatives such as Principal Component Analysis (PCA) or discriminant analysis are not as computationally efficient. The reduced data is likely to improve the Naive Bayes performance, but the CNN will require re-structuring to comply with the new input sequences. Since the sequences are aligned, many gap characters will diminish.

The CNN built in this work was based on the structure of LeNet-5 [31]. Due to the shape of the input data, a deep network with many convolution layers are not necessary if pooling quickly reduces the feature maps to a linear vector. However, altering the size and strides of the kernels may discover more meaningful spatial properties of the sequences. Furthermore, the optimal dropout probability may not be fixed and applied in this work. As more synthetic mutations are added to the training data, the train and test accuracy/loss diverge - indicating overfitting. To ensure the network's weights and biases do not solely explain the synthetic sequences, the dropout of some connected nodes may need to increase.

As elaborated in Chapter 1.2, statistical models have their merit in the species classification problem. Here, it has been shown that deep learning models may also add value to sequences classification, subject to memory limits. CNNs have their limitations: the pooling operation in CNN loses some features in the image and therefore require lots of training data in order to compensate for this loss. They also have long training times. A valuable study would be to apply other, more recent, deep learning techniques to this problem.

Few-shot learning, a sub-field of machine learning, is the problem of making predictions when the sample size is small [41] and can enable deep networks to handle cases with limited data. One can build a few-shot system by using Twin Networks [42]. A Twin Network is composed of two identically structured neural networks whose two outputs are fed into another function or network for a final result. If applied to these genetic sequences, the relationship of sequences belonging to the same species away from sequences belonging to different species is learned end-to-end. With the twin network, one generates an embedding for two sequences of the same species with the last layer performing a comparison through

a distance function. Twin Networks learns through Contrastive Loss, a loss function that minimizes when the two inputs belong to the same class, and maximizes when the two inputs belong to different classes. Applications of few-shot learning is found in speech technology, natural language processing and computer vision [43].

Capsule networks are also a new deep neural network architecture, which have shown a great performance in many fields, particularly in image recognition and natural language processing [44] [45]. Capsules are equivariant and are made up of a network of neurons which accepts and outputs vectors as opposed to CNNs' scalar values [46]. Translational invariance is particular to CNNs - they detect patterns in the input, whereas equivariance ensures that the spatial location of the patterns are taken into account. Capsule networks' ability to generalize well on smaller datasets makes them conducive for use in a wide variety of areas, as opposed to CNNs which must be trained and deployed on huge datasets.

There are many routes to pursue as a result of these analyses. A priority would be to consider COI sequences, to avoid using pre-classified data where assignment error and uncertainty with BINs are not present. Then, exploring up to date deep learning technologies that deal with small sampled data and high dimension data, as present in the BOLD databases.

Chapter 6

Conclusion

Current literature has eluded to the potential of statistical approaches to the species classification with genetic sequences, rather than standard similarity-based methods. Using aligned genetic sequences representing the *Insecta* class provided by LifePlan originating from the BOLD library, a Naive Bayes and CNN classifier were built. Using augmentation techniques inspired by the well-practice methods in image classification, the training sequences were expanded to improve generalisation. The results show that the CNN performs better than the Naive Bayes in all simulations for each data set considered. At the species level, the performance of the Naive Bayes model increases with the number of single-change synthetic mutations while the rate of improvement decreases. A similar outcome is observed in the results for the CNN. In particular, the training time to reach a maximum test accuracy decreases when more synthetic sequences are considered. A limit of the number of synthetic mutations beneficial to training was discovered, as a result of overfitting, which informed the final experiment. The optimal number of changes per synthetic mutations was estimate to be 5, at these hyperparameters the best performing model was identified with 76.4% test accuracy.

To conclude, whilst the results solidify the potential of using machine learning methods in classification, they also light a path to the use of deep learning methods. Further data augmentation techniques such as distortions, shifting and informed mutations are required to generalise the models. New deep learning techniques that perform well with low sample data such as few-shot learning and capsule networks will be a sensible extension to this project to enhance classification ability and hence our understanding of biodiversity in insects.

Bibliography

[1] Camilo Mora, Derek P. Tittensor, Sina Adl, Alastair G.B. Simpson, and Boris Worm. How many species are there on earth and in the ocean? *PLoS Biology*, 9, 2011.

[2] S. L. Pimm, C. N. Jenkins, R. Abell, T. M. Brooks, J. L. Gittleman, L. N. Joppa, P. H. Raven, C. M. Roberts, and J. O. Sexton. The biodiversity of species and their rates of extinction, distribution, and protection. *Science*, 344, 2014.

[3] Peter H. Raven and David K. Yeates. Australian biodiversity: Threats for the present, opportunities for the future. *Australian Journal of Entomology*, 46, 2007.

[4] Brett R. Scheffers, Lucas N. Joppa, Stuart L. Pimm, and William F. Laurance. What we know and don't know about earth's missing biodiversity. *Trends in Ecology and Evolution*, 27, 2012.

[5] Oskar L.P. Hansen, Jens Christian Svenning, Kent Olsen, Steen Dupont, Beulah H. Garner, Alexandros Iosifidis, Benjamin W. Price, and Toke T. Høye. Species-level image classification with convolutional neural network enables insect identification from habitus images. *Ecology and Evolution*, 10, 2020.

[6] Jan Bebbington, Thomas Cuckston, and Clément Feger. *Biodiversity*, pages 377–387. Taylor and Francis, 2021.

[7] Laura Klappenbach. How animals are classified. https://www.thoughtco.com/how-animals-are-classified-130745, 2019. (Accessed: 01.09.2021).

[8] A.J. Cain. Taxonomy. https://www.britannica.com/science/taxonomy, 2020. (Accessed: 01.09.2021).

[9] Paul D.N. Hebert, Alina Cywinska, Shelley L. Ball, and Jeremy R. DeWaard. Biological identifications through dna barcodes. *Proceedings of the Royal Society B: Biological Sciences*, 270, 2003.

[10] Sujeevan Ratnasingham and Paul D.N. Hebert. Bold: The barcode of life data system: Barcoding. *Molecular Ecology Notes*, 7, 2007.

[11] Sujeevan Ratnasingham and Paul D.N. Hebert. A dna-based registry for all animal species: The barcode index number (bin) system. *PLoS ONE*, 8, 2013.

[12] Howard A. Ross, Sumathi Murugan, and Wai Lok Sibon Li. Testing the reliability of genetic methods of species identification via simulation. *Systematic Biology*, 57, 2008.

[13] Frederic Austerlitz, Olivier David, Brigitte Schaeffer, Kevin Bleakley, Madalina Olteanu, Raphael Leblois, Michel Veuille, and Catherine Laredo. Dna barcode analysis: A comparison of phylogenetic and statistical classification methods. *BMC Bioinformatics*, 10, 2009.

[14] Emanuel Weitschek, Robin Van Velzen, Giovanni Felici, and Paola Bertolazzi. Blog 2.0: A software system for character-based species classification with dna barcode sequences. what it does, how to use it. *Molecular Ecology Resources*, 13, 2013.

[15] Mahzabeen Emu and Sadman Sakib. Species identification using dna barcode sequences through supervised learning methods. 2019.

[16] Qiong Wang, George M. Garrity, James M. Tiedje, and James R. Cole. Naïve bayesian classifier for rapid assignment of rrna sequences into the new bacterial taxonomy. *Applied and Environmental Microbiology*, 73, 2007.

[17] Gihad N. Sohsah, Ali Reza Ibrahimzada, Huzeyfe Ayaz, and Ali Cakmak. Scalable classification of organisms into a taxonomy using hierarchical supervised learners. *Journal of Bioinformatics and Computational Biology*, 18, 2020.

[18] Denan Xia, Peng Chen, Bing Wang, Jun Zhang, and Chengjun Xie. Insect detection and classification based on an improved convolutional neural network. *Sensors (Switzerland)*, 18, 2018.

[19] David Spiegelhalter. *Sex by Numbers : What Statistics Can Tell Us About Sexual Behaviour*. Profile Books Ltd., 2015.

[20] K. F. Tipton. Nomenclature committee of the international union of biochemistry and molecular biology (nc-iubmb). enzyme nomenclature. recommendations 1992. supplement: corrections and additions. *European Journal of Biochemistry*, 223, 1994.

[21] Phil Cunningham. Biological sequence analysis. probabilistic models of proteins and nucleic acids. *Cell Biochemistry and Function*, 17, 1999.

[22] Marco Sigovini, Erica Keppel, and Davide Tagliapietra. Open nomenclature in the biodiversity era. *Methods in Ecology and Evolution*, 7, 2016.

[23] Simon Greenhill. treemaker: A python tool for constructing a newick formatted tree from a set of classifications. *Journal of Open Source Software*, 3, 2018.

[24] Peter J.A. Cock, Tiago Antao, Jeffrey T. Chang, Brad A. Chapman, Cymon J. Cox, Andrew Dalke, Iddo Friedberg, Thomas Hamelryck, Frank Kauff, Bartek Wilczynski, and Michiel J.L. De Hoon. Biopython: Freely available python tools for computational molecular biology and bioinformatics. *Bioinformatics*, 25, 2009.

[25] Teuvo Kohonen. Median strings. *Pattern Recognition Letters*, 3, 1985.

[26] Bonnie Berger, Michael S. Waterman, and Yun William Yu. Levenshtein distance, sequence comparison and biological database search. *IEEE Transactions on Information Theory*, 67, 2021.

[27] M. R. B. Clarke, Richard O. Duda, and Peter E. Hart. Pattern classification and scene analysis. *Journal of the Royal Statistical Society. Series A (General)*, 137, 1974.

[28] Pat Langley, Wayne Iba, and Kevin Thompson. Analysis of bayesian classifiers. *Proceedings Tenth National Conference on Artificial Intelligence*, 1992.

[29] Simon Rogers and Mark Girolami. *A first course in machine learning.* Chapman and Hall/CRC, 2011.

[30] Vangelis Metsis, Ion Androutsopoulos, and Georgios Paliouras. Spam filtering with naive bayes - which naive bayes? *3rd Conference on Email and Anti-Spam - Proceedings, CEAS 2006*, 2006.

[31] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86, 1998.

[32] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. volume 2, 2012.

[33] Sebastian Ruder. An overview optimization gradients. *arXiv*, 2017.

[34] Steve Lawrence, C. Lee Giles, and Ah Chung Tsoi. What size neural network gives optimal generalization? convergence properties of backpropagation, 1998.

[35] Jason Brownlee. Dropout for regularizing deep neural networks. https://machinelearningmastery.com/dropout-for-regularizing-deep-neural-networks/, 2019. (Accessed: 26/08/2021).

[36] Frank Wilcoxon. Individual comparisons by ranking methods. *Biometrics Bulletin*, 1, 1945.

[37] Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7, 2006.

[38] Emanuel Weitschek, Giulia Fiscon, and Giovanni Felici. Supervised dna barcodes species classification: Analysis, comparisons and results. *BioData Mining*, 7, 2014.

[39] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63, 2020.

[40] Leland McInnes, John Healy, Nathaniel Saul, and Lukas Großberger. Umap: Uniform manifold approximation and projection. *Journal of Open Source Software*, 3, 2018.

[41] Yaqing Wang, Quanming Yao, James T. Kwok, and Lionel M. Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Computing Surveys*, 53, 2020.

[42] Wiley Wang, John Inacay, and Mike Wang. Few shot learning from scratch. https://deepganteam.medium.com/few-shot-learning-from-scratch-a3422b111e05, 2021. (Accessed: 26/08/2021).

[43] Maitreya Patel. Few shot learning - a case study (1). https://medium.com/analytics-vidhya/few-shot-learning-a-case-study-1-d71eb06a33df, 2020. (Accessed: 26/08/2021).

[44] Sara Sabour, Nicholas Frosst, and Geoffrey E. Hinton. Dynamic routing between capsules. volume 2017-December, 2017.

[45] Yongfeng Dong, Yu Fu, Liqin Wang, Yunliang Chen, Yao Dong, and Jianxin Li. A sentiment analysis method of capsule network based on bilstm. *IEEE Access*, 8, 2020.

[46] Mensah Kwabena Patrick, Adebayo Felix Adekoya, Ayidzoe Abra Mighty, and Baagyire Y. Edward. Capsule networks – a survey. *Journal of King Saud University - Computer and Information Sciences*, 2019.

Word Count: 7400