Bioinformatics Mini Project

The goal of this mini-project is to get each of you independently working with biological sequences performing real bioinformatics tasks at your own pace on your own computers and making use of your python skills. You should aim to put this project together in notebook format to collate your code and results. Please try and annotate your work and show intermediate steps and command-lines used. You may want to embed screenshots from external tools (e.g. Jalview) to show alignments etc. When coding, try and comment your code to assist yourself and us in understanding your methodology and rationale. For sections of the project that require command-line work, you can use command-line in *Jupyter* notebook with the exclamation symbol as shown in the practicals.

Part 1 – Open for Business - Open Reading Frame Detector in Python [25% marks]

It is somewhat of a tradition in bioinformatics that for many people the first program they write themselves without assistance is an 'Open Reading Frame' (ORF) prediction algorithm. This should be relatively simple in Python and will help you develop your skills in coding and dealing with sequences.

An Open Reading Frame (ORF) is a segment of DNA sequence that can encode for a protein. Firstly, the RNA polymerase enzyme will bind to the DNA and start to produce a messenger RNA through transcription. This piece of mRNA may contain instructions that encode a protein. A macromolecular machine called the Ribosome binds to mRNA and recognises the start codon, three nucleotides that trigger assembly of amino acids into a protein. The ribosome proceeds along from the start codon scanning three nucleotides at a time (codon triplets). Each three letter triplet encodes for specific amino-acids. Finally, when a specific stop codon triplet is detected the ribosome finishes assembly (translation) of a protein and the finished protein is released. During this process, amino acids are added in a codon specific manner by transfer RNAs (tRNAs).

The standard genetic code is the specific three letter triplet codons used by most organisms:

	ATA	I	CTA	L	GTA	V	TCA	S				
	ATC	I	CTC	L	GTC	V	TCC	S				
	ATT	I	CTG	L	GTG	V	TCG	S				
(Start Codon)	**ATG**	**M**	CTT	L	GTT	V	TCT	S				
	ACA	T	CCA	P	GCA	A	TTC	F				
	ACC	T	CCC	P	GCC	A	TTT	F				
	ACG	T	CCG	P	GCG	A	TTA	L				
	ACT	T	CCT	P	GCT	A	TTG	L				
	AAC	N	CAC	H	GAC	D	TAC	Y				
	AAT	N	CAT	H	GAT	D	TAT	Y				
	AAA	K	CAA	Q	GAA	E	**TAA**	*	(Stop codon)			
	AAG	K	CAG	Q	GAG	E	**TAG**	*	(Stop codon)			
	AGC	S	CGA	R	GGA	G	TGC	C				
	AGT	S	CGC	R	GGC	G	TGT	C				
	AGA	R	CGG	R	GGG	G	**TGA**	*	(Stop codon)			
	AGG	R	CGT	R	GGT	G	TGG	W				

Your first task is to develop a python program with the following goals:

1) Read in an DNA/RNA sequence file, ideally as FASTA format. You can write your own subroutine to read in standard FASTA format or you can use BioPython SeqIO as you prefer.

2) You may want to have your program be able to deal with both RNA and DNA sequences (i.e. handle U instead of T) and be able to handle variations in the FASTA format (e.g. some FASTA files have 60 characters before the sequence wraps to the next line, some 80 and some have no wrapping at all). Also you may have sequences with lowercase, uppercase and potentially non-standard characters.

3) Scan through the sequence to find open reading frames that start with an ATG and end with either TAA, TAG or TGA. This involves searching all possible 'frames' in the DNA sequence. For a DNA sequence this usually means searching three forward frames and three reverse frames that would correspond to the other strand of the DNA as it is double stranded so proteins could be produced from either strand.

As an illustration please consider the following DNA sequence we've been provided:

DNA Sequence Example:

Provided sequence:

```
ATTGCCATGACGGATCAGCCGCAAGCGGAATTGGCGTTTACGTACGGATGCGCCGTAATATAGGCCATAGAC
```

Three Forward Frames: (Triplet codes are alternating bold and regular font)

```
1:ATTGCCATGACGGATCAGCCGCAAGCGGAATTGGCGTTTACGTACGGATGCGCCGTAATATAGGCCATAGAC
2: TTGCCATGACGGATCAGCCGCAAGCGGAATTGGCGTTTACGTACGGATGCGCCGTAATATAGGCCATAG
3:  TGCCATGACGGATCAGCCGCAAGCGGAATTGGCGTTTACGTACGGATGCGCCGTAATATAGGCCATAGA
```

Work out Opposite DNA Strand (Complementary Base Pairs of original sequence)

```
ATTGCCATGACGGATCAGCCGCAAGCGGAATTGGCGTTTACGTACGGATGCGCCGTAATATAGGCCATAGAC
::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
TAACGGTACTGCCTAGTCGGCGTTCGCCTTAACCGCAAATGCATGCCTACGCGGCATTATATCCGGTATCTG
```

Reverse of opposite strand (reverse complement of original sequence):

```
GTCTATGGCCTATATTACGGCGCATCCGTACGTAAACGCCAATTCCGCTTGCGGCTGATCCGTCATGGCAAT
```

Three Reverse Frames:

```
4:GTCTATGGCCTATATTACGGCGCATCCGTACGTAAACGCCAATTCCGCTTGCGGCTGATCCGTCATGGCAAT
5: TCTATGGCCTATATTACGGCGCATCCGTACGTAAACGCCAATTCCGCTTGCGGCTGATCCGTCATGGCA
6:  CTATGGCCTATATTACGGCGCATCCGTACGTAAACGCCAATTCCGCTTGCGGCTGATCCGTCATGGCAA
```

4) This is not as straightforward as it looks, as the start codon (ATG) can start translation, but can also be present within a longer protein sequence. However, the stop codons will (almost) always stop translation. This is why, we usually search for the *longest* ORF within a sequence across frames.

5) In practice you will want to store the sequence and first iterate over the first three frames probably initially just printing out the triplets to make sure your code is working. You then need

to compute the longest distance between a stop codon (TAA, TAG, TGA) and any start codon TGA. For each frame store the longest detected ORF (if any) and record its start and end position.

6) Once complete you need to reverse complement your original DNA sequence to find the reverse strand sequence. In practice this can be done by reversing the DNA string and then changing each base to its complementary base (e.g. A -> T, G-> C, etc).

7) For the longest ORF found in all 6 frames you should use the standard genetic code above to work out the protein sequence produced by that reading frame.

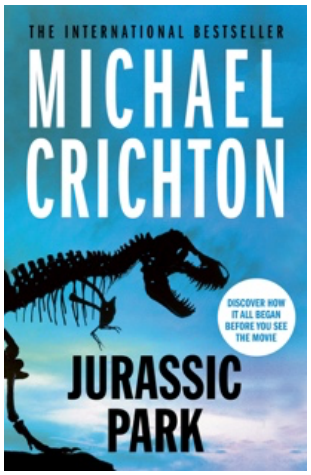
The example sequence above should have multiple ORFs, compute the longest amino acid sequence and store it as a FASTA file. You can also store shorter ORFs from other frames if you wish to explore them too.

While this initial project seems simple, there are many variations around ORF detection and it's also possible that you've been provided with an incomplete sequence where the start codon is not present. So try and think about all the possible variations that might occur and perhaps parameterise your program / subroutine so that the user can change how it runs.

8) You should download some longer human mRNA transcripts from Ensembl to test your implementation. An example human transcript for the Breas Cancer associated BRCA2 gene is here:
https://www.ensembl.org/Homo_sapiens/Transcript/Sequence_cDNA?db=core;g=ENSG00000139618;r=13:32315086-32400268;t=ENST00000380152

   Use the download sequence button to download the cDNA sequence (coding DNA) for the transcript. This is equivalent to the mRNA sequence after splicing before translation to protein. You should be able to identify the longest ORF and it should match the BRCA2 peptide sequence given in Ensembl.

Part 2: Jurassic Informatics – Decoding 'Dinosaur' DNA [25% marks]



In the 1990 book, Jurassic Park by Michael Crichton, scientists were able to produce dinosaurs by sequencing the DNA from fossils and used this to engineer embyros. Because the dinosaur DNA was degraded they allegedly filled in the gaps where DNA sequence wasn't available using frog DNA. On page 60 of the original book, there is a fragment of DNA sequence printed that purports to be a piece of dinosaur DNA reconstructed in this fashion.

Our goal here is to establish what exactly this sequence is (if anything) given that dinosaur DNA had not been sequenced in 1990 and indeed there were only 33,000 sequences comprising about 40 megabases at that time in the major databases.

```
>Dinosaur_DNA
GCGTTGCTGGCGTTTTTCCATAGGCTCCGCCCCCCTGACGAGCATCACAAAAATCGACGC
GGTGGCGAAACCCGACAGGACTATAAAGATACCAGGCGTTTCCCCCTGGAAGCTCCCTCG
TGTTCCGACCCTGCCGCTTACCGGATACCTGTCCGCCTTTCTCCCTTCGGGAAGCGTGGC
TGCTCACGCTGTACCTATCTCAGTTCGGTGTAGGTCGTTCGCTCCAAGCTGGGCTGTGTG
CCGTTCAGCCCGACCGCTGCGCCTTATCCGGTAACTATCGTCTTGAGTCCAACCCGGTAA
AGTAGGACAGGTGCCGGCAGCGCTCTGGGTCATTTTCGGCGAGGACCGCTTTCGCTGGAG
ATCGGCCTGTCGCTTGCGGTATTCGGAATCTTGCACGCCCTCGCTCAAGCCTTCGTCACT
CCAAACGTTTCGGCGAGAAGCAGGCCATTATCGCCGGCATGGCGGCCGACGCGCTGGGCT
GGCGTTCGCGACGCGAGGCTGGATGGCCTTCCCCATTATGATTCTTCTCGCTTCCGGCGG
CCCGCGTTGCAGGCCATGCTGTCCAGGCAGGTAGATGACGACCATCAGGGACAGCTTCAA
CGGCTCTTACCAGCCTAACTTCGATCACTGGACCGCTGATCGTCACGGCGATTTATGCCG
CACATGGACGCGTTGCTGGCGTTTTTCCATAGGCTCCGCCCCCCTGACGAGCATCACAAA
CAAGTCAGAGGTGGCGAAACCCGACAGGACTATAAAGATACCAGGCGTTTCCCCCTGGAA
GCGCTCTCCTGTTCCGACCCTGCCGCTTACCGGATACCTGTCCGCCTTTCTCCCTTCGGG
CTTTCTCAATGCTCACGCTGTAGGTATCTCAGTTCGGTGTAGGTCGTTCGCTCCAAGCTG
ACGAACCCCCCGTTCAGCCCGACCGCTGCGCCTTATCCGGTAACTATCGTCTTGAGTCCA
ACACGACTTAACGGGTTGGCATGGATTGTAGGCGCCGCCCTATACCTTGTCTGCCTCCCC
GCGGTGCATGGAGCCGGGCCACCTCGACCTGAATGGAAGCCGGCGGCACCTCGCTAACGG
CCAAGAATTGGAGCCAATCAATTCTTGCGGAGAACTGTGAATGCGCAAACCAACCCTTGG
CCATCGCGTCCGCCATCTCCAGCAGCCGCACGCGGCGCATCTCGGGCAGCGTTGGGTCCT
```

"And here is the revised DNA strand, repaired by the computer. The operation you've witnessed would have taken months in a conventional lab, but we can do it in seconds". "Then are you working with the entire DNA strand?" Grant asked. "Oh no," Wu said. "That's impossible. We've come a long way from the sixties, when it took a whole laboratory four years to decode a screen like this. Now the computers can do it in a couple of hours. But, even so, the DNA molecule is too big. We look only at the sections of the strand that differ from animal to animal, or from contemporary DNA. Only a few percent of the nucleotides differ from one species to the next. That's what we analyze, and it's still a big job."

Task:

Given that no dinosaur DNA was available in 1990 and even now dinosaur DNA is exceptionally difficult to obtain and or sequence, what exactly is this piece of DNA in Crichton's book?

1) Again, you will want to use your ORF finding python code to detect possible ORFs within the nucleotide sequence provided. Usually, we would take the longest open reading frame detected forward for subsequent searches after saving it as a FASTA file.

2) Using an online BLAST service (e.g. https://www.ebi.ac.uk/Tools/sss/ncbiblast/ at EBI or https://blast.ncbi.nlm.nih.gov/Blast.cgi?PAGE=Proteins at NCBI) search against annotated protein databases (*UniProt* or *nrdb*). Uniprot is a coordinated European collection of protein sequence annotations, while *'nr'* or *'nrdb'* is a similar database with redundancy removed (i.e. highly similar protein sequences are merged to one entry).

   As you are searching protein versus protein the appropriate tool is BLASTp. You can experiment with parameters to see if it alters your result.

3) Decide if this sequence is completely fictitious or whether it could be related to any real sequences present in the databases.

   You can also take the initial provided DNA nucleotide sequence and search this against nucleotide databases to see if it has any matches too.

Make sure to report your findings in your report/notebook with attached images or downloaded textual output from whichever server you choose.

4) For the sequel to Jurassic Park, Michael Crichton worked with a biologist (Mark Boguski) who'd analysed the first sequence (above) from the initial novel. Crichton then hired Mark to produce a (hopefully more convincing) DNA sequence for the next novel. That sequence is provided below (Dino_DNA_v2).

5) Redo the analysis with the sequence from the Lost World novel and decide if this is a more convincing piece of 'dinosaur' DNA and what exactly it may be based on. Are any of the hits from species related to dinosaurs?

   Remember to show and annotate your work (e.g. screenshots of alignment results or text based outputs) in your report.

Mr. DNA from the Jurassic Park Movie:

https://www.youtube.com/watch?v=qUaFYzFFbBU

>Dino_DNA_v2
GAATTCCGGAAGCGAGCAAGAGATAAGTCCTGGCATCAGATACAGTTGGAGATAAGGACGGACGTGTGGCAGC
TCCCGCAGAGGATTCACTGGAAGTGCATTACCTATCCCATGGGAGCCATGGAGTTCGTGGCGCTGGGGGGGCC
GGATGCGGGCTCCCCCACTCCGTTCCCTGATGAAGCCGGAGCCTTCCTGGGGCTGGGGGGGGGCGAGAGGACG
GAGGCGGGGGGGCTGCTGGCCTCCTACCCCCCCTCAGGCCGCGTGTCCCTGGTGCCGTGGGCAGACACGGGTA
CTTTGGGGACCCCCCAGTGGGTGCCGCCCGCCACCCAAATGGAGCCCCCCCACTACCTGGAGCTGCTGCAACC
CCCCCGGGGCAGCCCCCCCCATCCCTCCTCCGGGCCCCTACTGCCACTCAGCAGCGGGCCCCCACCCTGCGAG
GCCCGTGAGTGCGTCATGGCCAGGAAGAACTGCGGAGCGACGGCAACGCCGCTGTGGCGCCGGGACGGCACCG
GGCATTACCTGTGCAACTGGGCCTCAGCCTGCGGGCTCTACCACCGCCTCAACGGCCAGAACCGCCCGCTCAT
CCGCCCCAAAAAGCGCCTGCGGGTGAGTAAGCGCGCAGGCACAGTGTGCAGCCACGAGCGTGAAAACTGCCAG
ACATCCACCACCACTCTGTGGCGTCGCAGCCCCATGGGGGACCCCGTCTGCAACAACATTCACGCCTGCGGCC
TCTACTACAAACTGCACCAAGTGAACCGCCCCCTCACGATGCGCAAAGACGGAATCCAAACCCGAAACCGCAA
AGTTTCCTCCAAGGGTAAAAAGCGGCGCCCCCCGGGGGGGGGAAACCCCTCCGCCACCGCGGGAGGGGGCGCT
CCTATGGGGGGAGGGGGGGACCCCTCTATGCCCCCCCCGCCGCCCCCCCCGGCCGCCGCCCCCCCTCAAAGCG
ACGCTCTGTACGCTCTCGGCCCCGTGGTCCTTTCGGGCCATTTTCTGCCCTTTGGAAACTCCGGAGGGTTTTT
TGGGGGGGGGGCGGGGGGTTACACGGCCCCCCCGGGGCTGAGCCCGCAGATTTAAATAATAACTCTGACGTGG
GCAAGTGGGCCTTGCTGAGAAGACAGTGTAACATAATAATTTGCACCTCGGCAATTGCAGAGGGTCGATCTCC
ACTTTGGACACAACAGGGCTACTCGGTAGGACCAGATAAGCACTTTGCTCCCTGGACTGAAAAAGAAAGGATT
TATCTGTTTGCTTCTTGCTGACAAATCCCTGTGAAAGGTAAAAGTCGGACACAGCAATCGATTATTTCTCGCC
TGTGTGAAATTACTGTGAATATTGTAAATATATATATATATATATATATATCTGTATAGAACAGCCTCGGAGG
CGGCATGGACCCAGCGTAGATCATGCTGGATTTGTACTGCCGGAATTC

Part 3 – The Smell of Fear - Olfactory receptor Hunting in Mammalian Genomes [50% marks]

Olfactory receptors (ORs), also known as odorant receptors, are chemoreceptors expressed in the cell membranes of olfactory receptor neurons and are responsible for the detection of odorants (for example, compounds that have an odour) which give rise to the sense of smell. Activated olfactory receptors trigger nerve impulses which transmit information about odour to the brain. These receptors are members of the class A rhodopsin-like family of G protein-coupled receptors (GPCRs).

In vertebrates, the olfactory receptors are located in both the cilia and synapses of the olfactory sensory neurons and in the epithelium of the human airway. In insects, olfactory receptors are located on the antennae and other chemosensory organs. Sperm cells also express odour receptors, which are thought to be involved in chemotaxis to find the egg cell.

Your task in this part is to try and estimate the number and diversity of olfactory receptors in Human, Mouse, Dog and Cat.

You've been provided with an experimentally derived amino-acid sequence from a human olfactory receptor, below (detected_olfactory_protein).

```
>detected_olfactory_protein
MEKRNLTVVREFVLLGLPSSAEQQHLLSVLFLCMYLATTLGNMLIIATIGFDSHLHSPMY
FFLSNLAFVDICFTSTTVPQMVVNILTGTKTISFAGCLTQLFFFVSFVNMDSLLLCVMAY
DRYVAICHPLHYTARMNLCLCVQLVAGLWLVTYLHALLHTVLIAQLSFCASNIIHHFFCD
LNPLLQLSCSDVSFNVMIIFAVGGLLALTPLVCILVSYGLIFSTVLKITSTQGKQRAVST
CSCHLSVVVLFYGTAIAVYFSPSSPHMPESDTLSTIMYSMVAPMLNPFIYTLRNRDMKRG
LQKMLLKCTVFQQQ
```

Steps:

1) Our goal here is to build a HMM profile of olfactory receptors using the above sequence as our starting point. To find representative sequences for building our model, we will search this amino acid sequence above against all currently annotated human proteins to select a subset of high-scoring alignments of very likely olfactory receptors.

2) You should save the sequence above into a FASTA text file (.fasta extension) and then download all annotated human peptide sequences from ensembl: https://www.ensembl.org/info/data/ftp/index.html In this case you are after the 'Protein Sequence (FASTA)' file for Homo sapiens. That file should correspond to: *Homo_sapiens.GRCh38.pep.all.fa.gz*

   It is gzip compressed, so you may want to decompress it using *gunzip to: Homo_sapiens.GRCh38.pep.all.fa*

   Fasta files typically have *.fasta* or *.fa* as their extension, but are simple text files.

3) Download and install the NCBI BLAST package for your computer: https://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/

   There should be compiled versions for windows, macintosh and linux. You will need to use the commands: *makeblastdb* for the Human peptides you've downloaded. You will then want to

perform a protein-protein BLAST search using *blastp* of your amino acid sequence above against your formatted BLAST database. We did this during the practicals last term. This should generate a significant number of hits. You may find that the default output is useful for exploring the hits that have been returned visually, but that tabular output will be better suited to later processing (-outfmt 6). This output format allows you to specify specifically which bits of information will be useful. One example option would be:

-outfmt "7 qacc sacc qlen slen pident ppos length qstart qend sstart send evalue"

The above option will likely give you tabular output with the majority of information required for the next steps. The blastp command will list all the different output options.

It would also be possible to perform the BLAST search using online web-servers for BLAST, however the restricted range of output formats may not make your subsequent analysis easier. You might also prefer to use another fast heuristic dynamic programming tool such as FASTA from the FASTA package, (ftp://ftp.virginia.edu/pub/fasta) that's up to you, it shouldn't make a huge difference.

You should save your tabular alignment results and write a simple python program to do the following:

a) Check that each hit is significant (E-value <= 1e-30)
b) Check that the % similarity is >60%
c) Ensure that the alignment is close to full length e.g. > 290 amino acids. This is to avoid detecting 'pseudogenes' which are ancient remnants of genes which are no longer transcribed and not active. Hence they are free to mutate and drift. The human genome is littered with olfactory receptor pseudogenes.
d) Record which matched sequences (in the Homo sapiens database) satisfy these criteria.
e) Try and work out most likely human gene corresponding to the provided sequence.

4) Multiple Sequence Alignment

We will now use the sequences we've uncovered to build a multiple sequence alignment (MSA) across many olfactory receptors. This will allow us to explore the diversity of the sequences we've matched and allow us to build a Hidden Markov Model (HMM) trained on (hopefully) real olfactory receptors.

You should build a new FASTA file containing the sequences of the proteins identified in step three that passed your filter.

Ideally, you should choose the best 100 to 300 sequences from the BLAST search to build your alignment. The choice is up to you, too small and you risk building an alignment that doesn't cover the diversity of receptors and too many may be very slow for performing multiple sequence alignment.

Because this will be a very large MSA, you need to use one of the optimised clustering heuristic methods such as Clustal Omega or Muscle:

https://github.com/rcedgar/muscle/releases/tag/5.1.0
http://www.clustal.org/omega/

You could also load the FASTA file into Jalview (https://www.jalview.org) which has the ability to load an unaligned FASTA file and perform web-based alignment via many methods.

5) MSA Analysis

You may have performed alignment on the command line using MUSCLE or CLUSTAL Omega, in which case you should choose an output format (e.g. MSF) that you can load into Jalview for viewing. We demonstrated generating alignments during the practicals last term, but here we will be doing a lot of interactive analysis of the alignment we have obtained.

If you generated your alignments using Jalview's web-services then they will already be displayed in a new window.

You should look over your alignment carefully, you may have some sequences that crept in that are not full length or contain large insertions or deletions. These can be selected and removed from the alignment.

You should use Jalview's option to generate a phylogenetic tree from your alignment. Using **Calculate→Calculate Tree or PCA** in Jalview will build a Phylogenetic Tree based on the alignment. You can build both Neighbour Joining or a UPGMA based tree.

Using either the Tree you've built or even PCA analysis of the alignment explore the alignment and see if you can see sub-classes of olfactory receptors. You can go back to Ensembl and see if the annotations of the proteins match the sub-classes you are observing from either Tree or PCA analysis.

You should document the tree and/or PCA results and your findings in your notebook.

6) Structural features in the alignment

Looking over the alignment you should see more highly conserved regions where there is less sequence difference. These are likely the key regions of the protein for their function. See if you can annotate these regions based on looking over your sequence alignment. Changing the colouring on the alignment may help you spot those conserved domains that are the key determinants of the type of olfactory receptor.

7) Building a HMM for your alignment

Download and install HMMer (http://hmmer.org/download.html) . On a macintosh you can likely use homebrew (https://brew.sh) to have it download a pre-compiled binary version rather than build it from scratch.

Use *hmmbuild* to generate a new *.hmm* file containing a HMM profile for your aligned sequences.

8) Exploring mammalian olfactory receptor diversity

Download a set of mammalian peptides from Ensembl, similarly to what you did to obtain the human peptides. You may want to try: Mouse (*mus musculus*), Dog (*Canus lupus familiaris*) and Cat (*Felis catus)* but feel free to go nuts and add some weird species to your analysis but you will likely do best sticking to mammals.

Using *hmmsearch* you can compare your HMM profile across all the peptides from each species to estimate the number of olfactory receptors. You should save the output from each search and use python to parse the output to select matches which are full length and sufficiently convincing (e.g. using E-Value).

Collate your results and make a plot of the number of receptors you can detect in each species.

One possible problem which may affect your counts is that due to how Ensembl annotates peptides you may see a lot of redundancy where multiple, identical or very similar peptides are present in the FASTA file which originate from the same gene. This could be inflating your counts in species which are better annotated.

Think of a strategy to improve your estimate of the olfactory repertoire in each species by not double counting hits to peptides from the same genomic position or gene. You can describe these in your notebook, but no need to go and try them for this mini-project.

Notes:

Our approach here is not entirely ideal. Much depends on how good our initial alignment was and we may easily miss rare olfactory proteins that didn't make it into our initial alignment. We are also likely overtraining because some classes will be more abundant in our alignment compared to others and our starting alignment is human only. However, we should still be able to obtain a reasonable estimate of olfactory repertoire and also have a reasonable understanding of the limitations of our analyses and perhaps how to do it better.

If you are interested in exploring this further, you could visit HORDE from the Weizmann institute where Human olfactory receptors are annotated in great detail.
https://genome.weizmann.ac.il/horde/

You could, if you wished build (better) MSA alignments based on this, or even subclass specific HMMs.