RESEARCH NOTES ON MACHINE OLFACTION

Collecting the datasets, understanding the content and quality of each dataset, ex. what variables are the features and which ones are potential labels

Step 1

Dataset 1: MNIST dataset — can biological neural networks be applied to an ML context? Can you outperform classic ML models on recognizing the MNIST dataset of handwritten digits using a BNN (The Moth olfactory network)?

Not really an olfaction dataset, just something to show how adaptable BNN's are.

Dataset 2: DREAM olfaction prediction challenge

- 476 structurally and perceptually diverse molecules
- 4884 physicochemical features of each of the molecules smelled by the subjects, including atom types, functional groups, and topological and geometrical properties that were computed using Dragon chemoinformatic software.

Predictions that can be made:

- Perceived intensity (rating the amount a molecule smelled of a particular odor descriptor on a scale of 1-100), pleasantness, usage of the 19 semantic descriptors for each of the 49 individuals and for the mean and standard deviation across the population of these individuals

Challenges:

- There are a lot of features, picking the minimal set of features that give you a good prediction is key. Feature selection methods are going to be important.

- Joining the datasets looks to be a big task. The different text files seemingly contain different features, and there's going to be a lot of pre-processing work to be done before applying any kind of ML algorithms
- There are a lot of CSV/text files in the data folder, and the Readme isn't very informative and descriptive about the contents of each text file.

Dataset 3: The paper that uses Graph Neural Networks assembled an expert-labeled set of 5030 molecules from two separate sources: the GoodScents perfume materials database (n = 3786, [44]) and the Leffingwell PMP 2001 database (n = 3561, [45]). The datasets share 2317 overlapping molecules. Molecules are labeled with one or more odor descriptors by olfactory experts (usually a practicing perfumer), creating a multi-label prediction problem.

Neither of those databases look to be freely available.

Dataset 4: Sensory perception of 480 different molecules at two different concentrations as experienced by 55 demographically diverse healthy human subjects. Subjects rated intensity, pleasantness, familiarity, and applied 20 odor descriptors. Two different dilution levels (1/1000 and 1/100,000.

Can you predict the molecule based on all of the other factors?

Dataset 5: https://www.flavornet.org/index.html has odorants, their corresponding odors, and their molecular weights. Some molecules can have multiple odors associated with them.

There is also a page with the list of molecules associated with each unique odor, and another page odor classes (Animal, aromatic, berry etc.).
This may not be enough on its own to make any predictions, but it could be added to other datasets to improve their quality.
One problem is that this is not a downloadable dataset, so we are going to need a web scraping script to get it.

***** Need to use something like RDKit, Open Babel or the Dragon Software to generate chemoinformatic features for the last two datasets. *****

Collect relevant papers, read them and understand how different ML methods are applied to the problem, their accuracy/practical applications.

Step 2

Paper 1: Insect Cyborgs: Bio-mimetic Feature Generators Improve Machine Learning Accuracy on Limited Data (**https://arxiv.org/pdf/1808.08124.pdf**)

Goal: To auto-generate stronger feature sets, to aid ML methods faced with limited training data.

- They use MothNet as an automatic feature generator to derive new features from the original ones that can be used by standard ML algorithms.
- On the MNIST data set, this method outperforms both applying ML methods on the original features and other feature generating methods like PCA, PLS and NN's.

- Problems with ML methods: they need large amounts of training data to attain high performance.
- The readout neurons of MothNet (AL-MB) contain class separating information that will boost an arbitrary ML algorithm's ability to classify test samples.
- The original input features (pixels) of the vectorized MNIST and Omniglot data contain class-relevant information which is not accessed by ML methods, but which the MothNet encodes in a form that is accessible to the ML methods.

Paper 2: Olfactory perception of chemically diverse molecules (**https://bmcneurosci.biomedcentral.com/articles/10.1186/s12868-016-0287-2**)

Goal: To understand the relationship between a stimulus and how it is perceived.

- Is there a relationship between the molecular features of the stimulus and olfactory perception?
- This dataset is more geared towards improving the quality of datasets used to predict pleasantness and intensity of an odorant, rather than actually predict those things.
- Key conclusions include gathering correlations based on bullet #1

Paper 3: Predicting human olfactory perception from chemical features of odor molecules (**https://science.sciencemag.org/content/355/6327/820**)

Goal: To predict sensory attributes (like intensity, pleasantness, semantic descriptors) of a molecule using their chemoinformatic features.

- It is not currently possible to predict the smell of a molecule from its chemical structure.
- Using dataset #2, teams made models to relate chemical structure to odor percept.
- Random forest and regularized linear models outperformed other common predictive model types for the prediction of individual and population perception.
- Some attributes required hundreds of features to be optimally predicted, random forest and linear models achieved an 80% prediction quality with far fewer features.

Maybe solve how pleasantness and intensity relate to what the descriptor is?

Paper 4: Diagnostic Value of the Impairment of Olfaction in Parkinson's Disease (**https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3655992/)**

Goal: To find out whether odor identification can be a supportive diagnostic tool for Parkinson's disease.

- Odor identification was assessed with 16 Sniffin' sticks in 148 Parkinson patients and 148 healthy controls.
- Risks of olfactory impairment were estimated with proportional odds models, and Random Forests were applied to classify the patients into Parkinson's and non-Parkinson's.
- Results showed that Parkinson patients' sense of smell was rarely normal, and age related impairment of olfaction showed a steeper gradient in Parkinson patients.
- Resuts indicate that testing odor identification can be a supportive diagnostic tool for Parkinson's disease.

Step 3

Packages/Software/ML Methods and challenges involved:

- The Dragon Software is the most popular method to calculate molecular descriptors. It calculates 5270 descriptors, covering most of the various theoretical approaches; but it is paid.
- The DREAM olfaction challenge (Paper 3) provided 4884 physicochemical features that were calculated using the Dragon software.
- From the above papers, classic ML methods can be applied to machine olfaction (linear models, random forests, PCA, NN's for feature generation)
- The big challenge is feature selection: a model that needs fewer features is more desirable, and therefore being able to generate stronger feature sets (perhaps by using MothNet?) is key.
- DeepChem is an open-source deep learning software that "democratizes the use of deep-learning in drug discovery, materials science, quantum chemistry, and biology." Potentially something I can use if I want to be able to apply deep learning to my datasets.
- ChemProp contains message passing neural networks for molecular property prediction — probably not as useful in this case, as the datasets already have molecular properties calculated using Dragon Software. Definitely useful when building new datasets.
- MoleculeNet is another package that provides benchmarks for ML methods that predict molecular properties.

FLAVORNET DATASET RESULTS

Data visualization:

- To get a sense of the distribution of the number of descriptors that a molecule has (Fig 1)

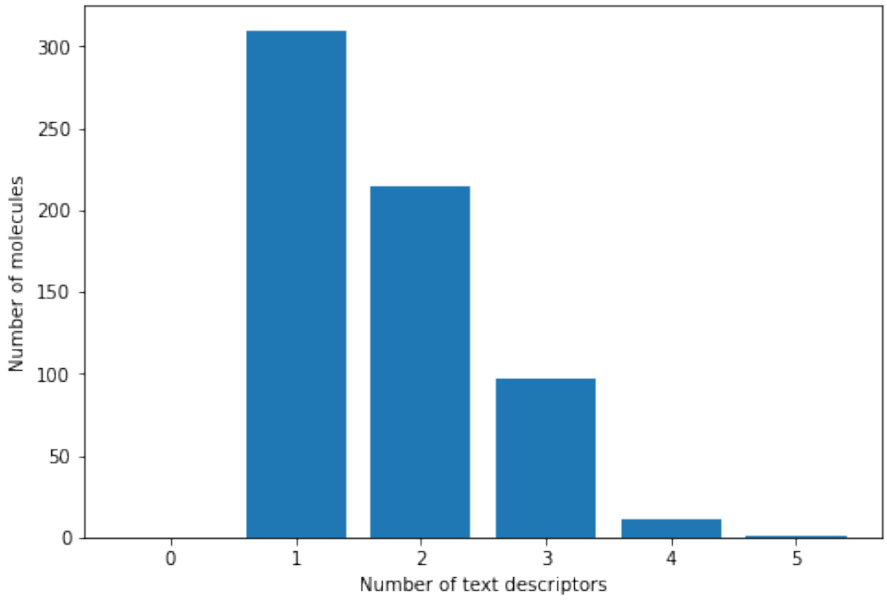


Fig 1

- Graph that displays connection strength between pairs of descriptors that appear for the same molecule more than 3 times; the darker the edge the more times they appear together (Fig 2)
- There's not a lot of frequent pairs — could indicate that correlation between descriptors isn't significant (probably due to the large number of descriptors), and the multi-class prediction algorithm should be chosen appropriately.

- Lots of pairs of descriptors were dropped, which indicates another challenge for prediction: how to evaluate correctness? Is it correct if the algorithm predicts just one of the descriptors?

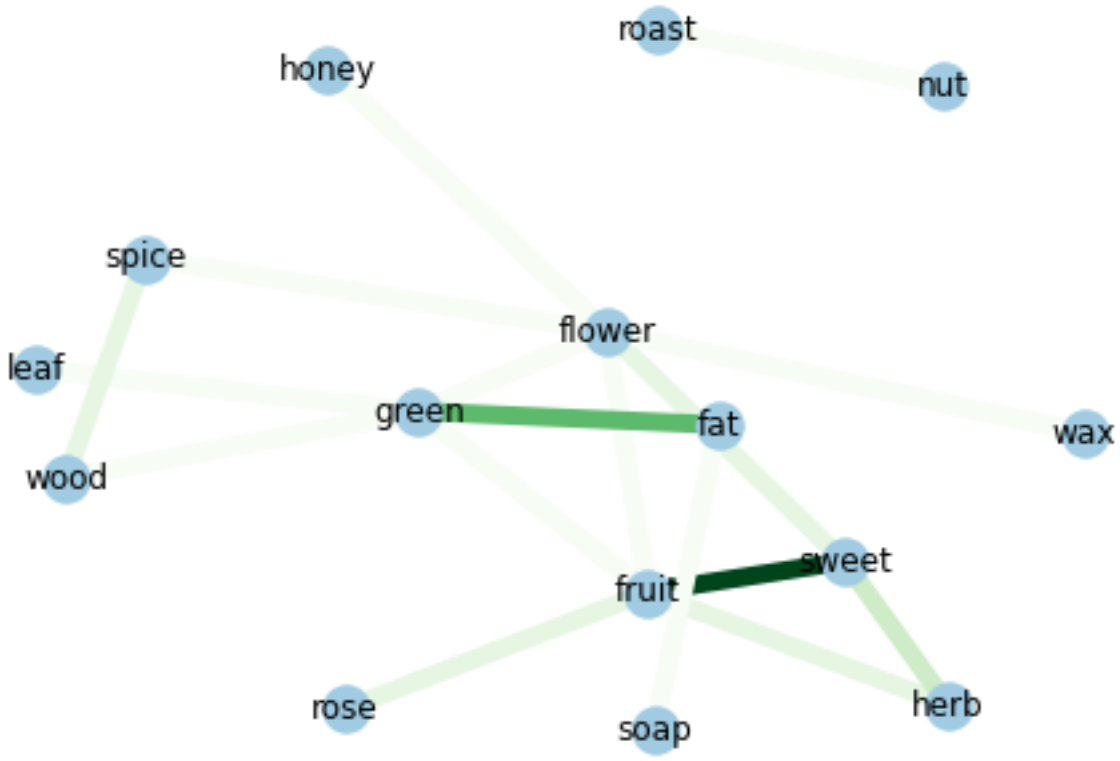


Fig 2

- The correlation heat map (Fig 3) is a more intuitive way of visualizing the relationship between the 15 descriptors that appear together at least thrice in the dataset.
- Created it by finding the normalized laplacian matrix of the graph in Fig 2, and the resulting matrix is symmetrical.
- Idea for using the normalized laplacian matrix for correlation is from **https://arxiv.org/pdf/1511.07953.pdf**.

- It nearly mimics the relationships found in the connection strength graph, and finds some new ones: leaf and green have significant correlation, which is not something you can see in Fig 2.

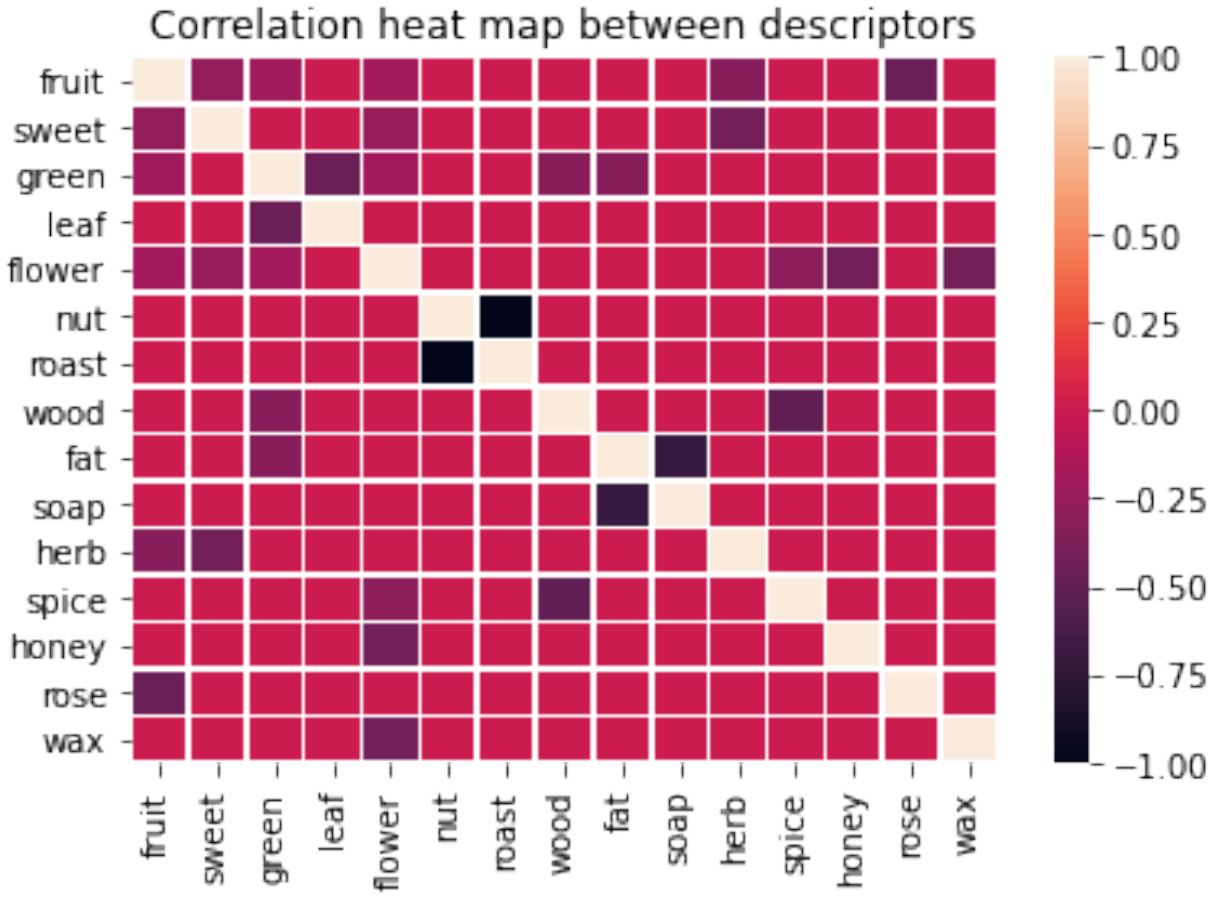
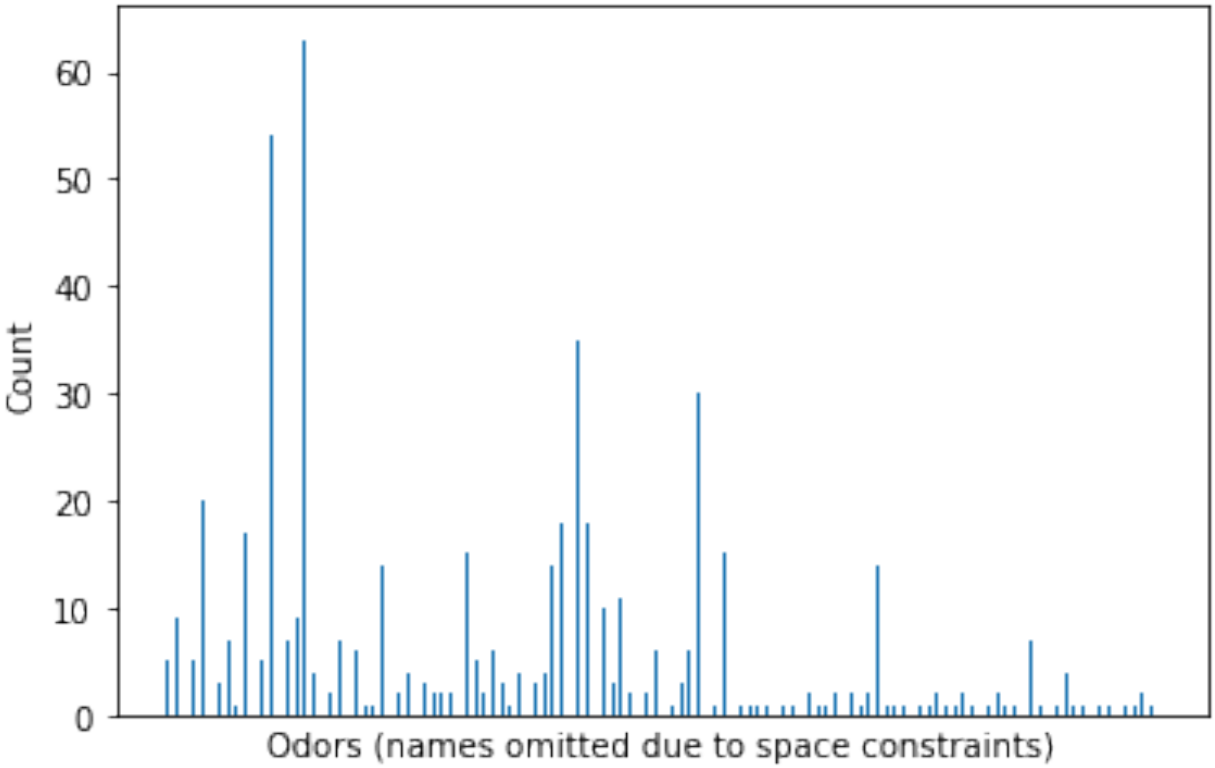


Fig 3

Class imbalance

Imbalanced classes can lead to poor performance in the classification algorithms. To determine it, I plotted a bar graph of all the odors and the number of times they appear in the dataset



- The discrepancy between the odor descriptor that appears the most (fruit, 63) and the least (many odors, 1) is very large — clear case of class imbalance
- Downsampling methods are not viable as dropping the most frequent labels could result in associated labels being dropped as well.
- Upsampling methods will give us multiple labels with a similar pattern, making the dataset prone to overfitting.

MODELS:

RANDOM FORESTS

Model number	Parameters	Hamming Loss	Precision (across all samples)	Recall (across all samples)
1	class_weight='balanced'; **criterion='gini'**	0.010823	0.26756	0.11539
2	class_weight='balanced'; **criterion='entropy'**	0.010756	0.28529	0.12615
3	Default parameters	0.010389	0.32061	0.12682
4	Default parameters + 10-fold CV	0.010395	0.30672	0.12115

- Very low hamming loss is good, indicates a small number of false positives and false negatives
- But the precision and recall are low. This tells us that because there are a majority of zeros as labels for a sample, the model errs too much on the side of "safety", predicting the absence of an odor.
- Altering the value of 'k' in k-fold cross validation does not change the result significantly; precision and recall are still too low.

With basic AlvaDesc properties:

Model number	Parameters	Hamming Loss	Precision (across all samples)	Recall (across all samples)
1	class_weight='balanced'; **criterion='gini'**	0.01184	0.2363	0.13796
2	class_weight='balanced'; **criterion='entropy'**	0.01195	0.21098	0.11882
3	Default parameters	0.01124	0.3011	0.1344
4	Default parameters + 10-fold CV	0.01031	0.3124	0.1222

CONCLUSION:

- Random forests are not good enough because they don't predict "positive" class labels well enough, which is more important to do in this context.

DEEP NETWORKS

- With two hidden layers, the recall jumps significantly when compared to random forests, precision goes down and hamming loss rises as well. Recall score is pretty good but still problematic because high hamming loss indicates it's probably due to it predicting a lot more 1's than random forests. **Recall** 18.7%, **Hamming loss** 22%, **Precision** 1.2%
- Even by dropping labels that appear only once in the dataset (there are 75) of them, the results remain approximately the same. It's looking like the problem is with the features, they do not seem to be good enough.
- Adding more layers/neurons won't help, it will just lead to overfitting. Maybe we can increase the threshold so more labels are dropped, but that is unlikely to help.

With basic AlvaDesc properties,

- Recall shoots up to 44.5%, hamming loss is at 33.7% (numbers from just one run), precision is at 1.1%. Significant increases in recall (good), hamming loss (bad), precision remains the same.
- Increase in recall is > increase in hamming loss