CS 152 Project: Checkpoint 1

Nathaniel Davis
Tufts University

Kamil Krukowski
Tufts University

Yan Zhou Chen
Tufts University

Abstract

Purpose: In this project we predict the mass spectra of molecules from their molecular fingerprints, a task which, due to the sparseness of spectrum data, we approximate by predicting which discretized spectrum "bin" contains the highest-intensity peak. This task is useful as metabolite annotation via mass spectrometry requires reference spectra for comparison; however, we lack reference spectra for millions of recorded molecules, so in-silico spectra generation helps fill these gaps. **Hypothesis:** We use feedforward neural networks with SoftMax output to generate probabilities that a given bin contains the highest peak, and to approximate the posterior of a network's weights, we use Bayes-by-Backprop (BBB) with a multivariate Gaussian posterior approximation with three different variance assumptions: uniform variance with no covariance, independent variances with no covariance, and per-layer independent variances with covariance. For each approach, we measure average SoftMax rank of the correct bin, top-k accuracy, and calibration of bin probabilities, as well as a runtime analysis involving time to model convergence. **Evaluation Plan:** We use liquid chromatography tandem mass spectra (LC-MS/MS) data from the MassBank of North America (MoNA), a large (145k spectra) public dataset, to evaluate our models by the metrics mentioned above.

1 Changelog

- Instead of treating the problem as a vector regression task, we instead attempt to predict which discretized spectrum bin contains the highest-intensity peak. Most spectrum bins contain values at or near zero, so most of any error/closeness metric is generated by whichever bin(s) contain(s) the highest peak(s). Therefore, if we can predict the bin with the highest peak with decent accuracy, we are well on our way to satisfying most objective metrics for the original vector regression task.
- To evaluate the probabilistic predictions of our models, we measure prediction calibration instead of out-of-distribution detection. The former gives us a more precise, intuitive sense of the Bayesian capabilities of our models and also does not require crisply split in-distribution and out-of-distribution sets. Molecular fingerprints are very inflexible representations based on in-distribution data and therefore completely 'drop' out-of-distribution information.
- We listed a host of models we intended to test in our Project Pitch, but we have now limited our approach to BBB, a more realistic problem scope given our timeline. We have also decided to implement a vanilla, non-Bayesian neural net to provide baseline estimates of non-Bayesian performance metrics.

2 Challenges

- Given the size of our inputs/outputs, implementing BBB with per-layer full covariance matrices may prove computationally intractable.

- The SoftMax framing of the problem, coupled with the fact that molecular fingerprints ignore all spatial information, may not produce results of reasonable quality.

3 Timeline

By Monday, 11/14

- Implement vanilla neural network and decide hyperparameters - Yan
- Decide and implement validation strategy - Kamil
- Review academic references and code for focus method and baseline - Nate

By Mon 11/21

- Implement and evaluate uniform variance BBB model - Yan & Nate
- Implement and evaluate BBB variants - Kamil
- Compare model performance and begin checkpoint write up - everyone

4 Goals

In-silico mass spectra are frequently used as references to help annotate experimentally produced spectra, yet the uncertainty of spectra predictions is rarely reported. We aim to approximate a molecule's mass spectrum by predicting the bin location of its highest-intensity peak given its fingerprint while reporting the uncertainty of the model. Having uncertainty estimates should aid in the process of matching experimentally produced and synthetic spectra. We model this as a multiclass classification task with a training set of observations: $\{x_{1:N}, y_{1:N}\}$ where each input is a molecular fingerprint of size 166 and each output y_n is a one-hot encoding of size 1,000. We develop a vanilla feed-forward neural network and several BBB models with multivariate Gaussian posterior approximations of varying flexibility. We expect our dataset features to have high covariance, suggesting that perhaps the hidden layers should also have similarly high covariance. For this reason, we expect posterior approximations with fixed variance and independent weight variances to perform worse than the non-Bayesian neural network. However, we expect the posterior approximation with per-layer weight covariance to capture these dependencies and outperform the vanilla neural network

5 Methods

5.1 Model Objectives

Our probabilistic model consists of random variables θ for weights and biases of the feed-forward neural network, $f(x, \theta)$ and y for the outputs of prediction. We assume a Gaussian distribution over θ as its prior (1). Specifically, we plan to use a normal prior with $\mu = 0$ and $\Sigma = I$. The likelihood is a categorical distribution with parameters $\pi_1, ..., \pi_K$ computed from the feed-forward neural network (2). We describe the model posterior in (3).

$$p(\theta) = \mathcal{N}(\mu, \Sigma) \tag{1}$$

$$p(y|\theta) = \text{Cat}(\pi_1, ..., \pi_K) = \text{Cat}(f(x, \theta)) \tag{2}$$

$$p(\theta|y_{1:N}) = \frac{p(y|\theta)p(\theta)}{p(y_{1:N})} \tag{3}$$

The posterior is approximated using Gaussian mean-field VI. First, we define a distribution $q(\theta; m, r(s))$, where each θ_j is independently normally distributed with mean m_j and variance $r(s_j)^2$, where r represents the softplus function (4). We aim to minimize the the KL divergence from $q(\theta; m, r(s))$ to $p(\theta|y_{1:N})$, or equivalently, maximize the ELBO, defined as $\mathcal{L}(m, s)$ (5). Using the reparameterization trick, we define a new function $T(m_j, s_j, \epsilon_j)$ (6) and derive the gradient of the ELBO with respect to m(7). The derivative with respect to s can be defined in a similar way.

$$q(\theta; m, s) = \prod_{j=1}^{J} \mathcal{N}(\theta_j|m_j, r(s_j)^2) \tag{4}$$

$$\mathcal{L}(m, r) = \mathbb{E}_q[\log p(y, \theta) - \log q(\theta; m, r)] \tag{5}$$

$$\epsilon_j \sim \mathcal{N}(0, 1); \theta_j \leftarrow m_j + r(s_j)\epsilon_j = T(m_j, s_j, \epsilon_j) \tag{6}$$

$$\nabla_m \mathcal{L} = \mathbb{E}_{\epsilon \sim \mathcal{N}(0,1)}[\nabla_m \log p(y, T(m, s, \epsilon)) - \nabla_m \log q(T(m, s, \epsilon))] \tag{7}$$

In addition to Gaussian mean-field VI with independent weights and biases, we plan to explore the model's performance with fixed variance σ^2 scheme, that is $r(s_j)^2 = \sigma^2$ for all j in equation (4) and per-layer covariance scheme where (4) becomes (8) for a network of L layers, where Σ^l is a JxJ matrix for layer l. We believe this covariance variable will help us achieve our goal best among all methods. The performance of the vanilla neural network serves as the baseline for the three proposed bayesian models and sanity check for Pytorch.

$$q(\theta; m, s) = \prod_{l=1}^{L} \mathcal{N}(\theta^l | m^l, \Sigma^l) \tag{8}$$

5.2 Calibration

Our model derives a posterior predictive over our prediction.

$$p(y|x) = \int p(y|x, \theta) p(\theta|x) d\theta \tag{9}$$

This allows us to derive a confidence interval for our prediction with a lower and upper bound $\hat{y}_l, \hat{y}_u$. We define this in terms of a confidence function that takes in a parameter p referring to the interval width.

$$(\hat{y}_l, \hat{y}_u) = \text{C}(x, p) \tag{10}$$

Where we expect

$$P[\hat{y} \in (\hat{y}_l, \hat{y}_u)] = p \tag{11}$$

We apply a post-training affine correction to the uncertainty of our model. We call this correction T.

$$(m_1\hat{y}_l + b_1, m_2\hat{y}_u + b_2) = T(x, p) \tag{12}$$

We learn m_1, m_2, b_1, b_2 through the following objective using gradient descent. We expect our predicted interval corresponding to p to correctly capture p of the data. Let $L(p)$ represent our "L1 Calibration Loss".

$$L(p) = p - \mathbb{E}[\![\hat{y} \in T(x, p)]\!] \tag{13}$$

We square our loss to derive a "Mean Square Error" or "L-2" training objective, and integrate across all interval widths.

$$\arg\min_{m,b} \int ||L(p)||^2 dp \tag{14}$$

For the sake of computation, the expectation in our loss and the integral are both intractible – We will approximate both using Monte-Carlo methods over the validation set V of samples and a set S of uniformly distribution p values.

$$\arg\min_{m,b} \frac{1}{|S|} \sum_{p \in S} [p - \frac{1}{|V|} \sum_{v \in V} \hat{y_v} \in T(x_v, p)]^2 \tag{15}$$

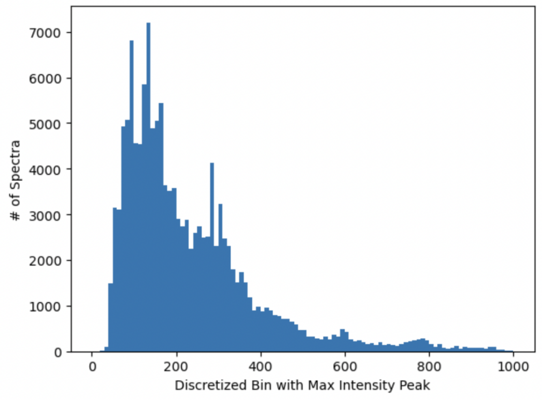
6 Experiments

6.1 Dataset

We plan to use MoNA (MassBank Of North America), an online database of mass spectrometry experiments, for our data. Specifically, we will use the LC-MS/MS subset of approximately 145,000 unique spectra. The data is public, so we have already accessed and begun to process it. We chose this dataset as it was the largest publicly available one we could find. (*Note*: There is no paper associated with the database.)

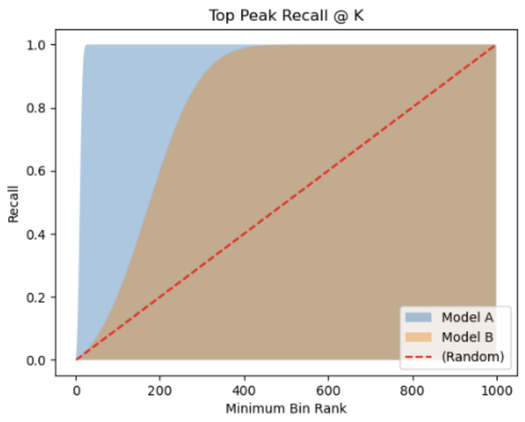
The two key fields for every data point are the molecular fingerprint and spectrum representation. Our fingerprints are binary vectors of length 167, in which each feature indicates the presence (1) or absence (0) of a specific molecular substructure. Mass spectra generated by spectrometers are continuous, but for the purposes of analysis, the spectra representations we use are vectors of 1000 discretized bins. The bins are increments of 1 Dalton (a molecular unit of mass), containing the max (positive) intensity value recorded within its range.

The spectra within our dataset have all been generated from organic molecules and via liquid chromatography tandem mass spectrometry (LC-MS/MS), a mass spectromety technique especially well suited to small molecules. The spectra have been generated from approximately 10,000 unique compounds. Lastly, as we plan on predicting the bin with the highest-intensity peak, we have produced the distribution of these bin locations below. Note that max peak locations are strongly skewed toward lower mass/charge bin values, though they do run the full range.



6.2 Intended Experimental Results

Visualized below are toy examples of two graphs we wish to plot for our given models.



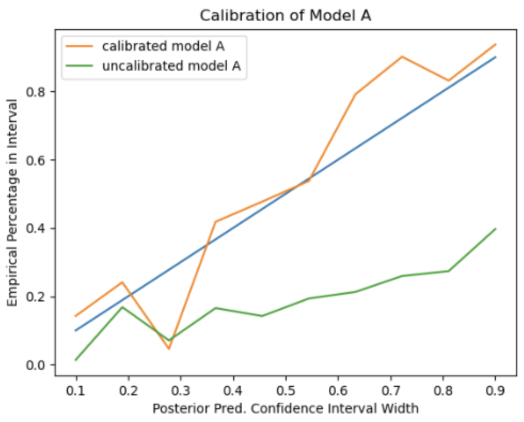


Figure 1 (left): The first key plot will be a recall curve for the highest intensity bin across our models. This will involve overlapping three histograms, one for each model, indicating the number of test spectra whose correct highest-peak bin falls within the top K bins.

Figure 2: (right): Our second key plot will compare the calibration of our BBB models with different posterior variance assumptions. We calculate calibration as the difference between the expected number of peaks that fall within our confidence intervals according to our model and the actual number of peaks witnessed without our confidence intervals from test-set experimental data. An ideal model shows 1-1 correspond or 'diagonal line' on this plot.

6.3 Implementation Plan

For implementation, we plan to use PyTorch. Each team member has prior experience with it from previous research. We each have past examples of implementing vanilla neural networks in PyTorch, so getting ours up and running should not take too much work. We will have to write PyTorch versions of BBB from scratch. We can use the JAX version of BBB from our homework, however, as a guide. We plan to experiment with Pytorch Lightning to gain some experience in a different styled framework. We plan to experiment with Optuna to gain experience in a hyperparameter optimization framework and help automate parameter selection.