LC/MS Chromatogram Smoothing With Generative Adversarial Networks

Gabriel Ferns(CS Senior) , Max Horowitz-Gelb(CS Senior)

May 11, 2017

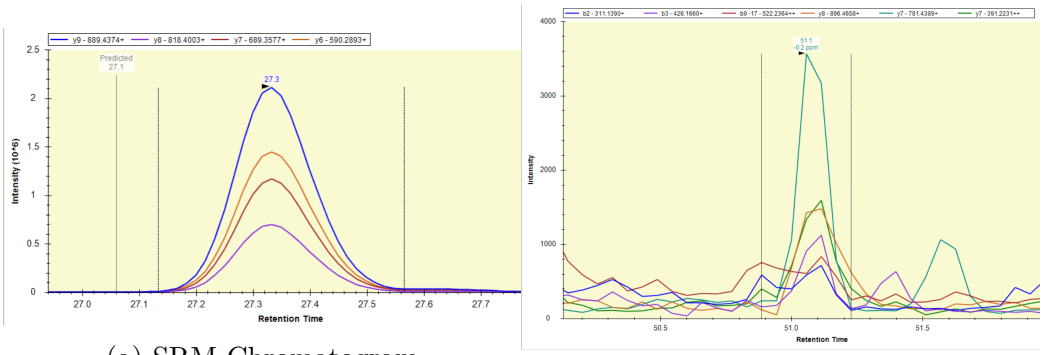
Abstract

Liquid Chromatography Mass Spectrometry is a powerful tool for identifying and quantifying proteins in a complex sample. High throughput methods for LC/MS allow allow for analysis of samples with thousands of proteins. But this increase in throughput comes at the cost of a decrease in signal quality. This issue calls for post extraction methods to clean and smooth chromatogram signal data. Here , using a new and powerful deep learning framework known as GANs [4], we attempt to smooth chromatograms, without distortion of the true underlying quantifiable data. From our experiments we show promising results which are able to somewhat achieve the same shape as our input while not distorting the quantifiable data.

1 Introduction

DIA [3] is a method of LC/MS for quanitifying thousands of proteins in a single sample. In order to analyze thousands of proteins at the same time, DIA requires a low sample rate. This results in the quality of the data extracted to be quite poor. An alternative to DIA is SRM, or Selected Reaction Monitoring, [8]. SRM is not good for quantifying more than a handful of peptides at a time, but the data generated is of far superior quality. It would be nice if we could keep the high throughput of DIA while also having data with the same quality as SRM. Then one could analyze thousands of proteins at the same time and get clean reliable data for quantification.

The quality of SRM data in comparison to DIA data suggests that



(a) SRM Chromatogram

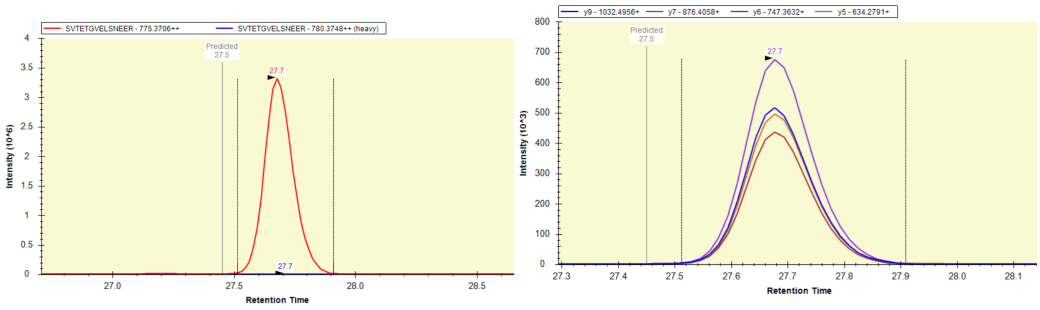
(b) DIA Chromatogram

Figure 1: Here we show two chromatograms. As one can see the noise and the smoothness is much better in the chromatogram coming from an SRM dataset.

perhaps a generative model trained on clean SRM data, could remove noise and smooth DIA data without distorting the actual quantifiable information. To test this hypothesis we chose to use a GAN as our generative model. GAN neural networks have been shown as powerful tools for image processing particularly for removal of noise and upscaling of images [5] [12]. We applied similar methods to chromatogram data by training a GAN on clean SRM dataset with simulated Gaussian noise. We then tested our GAN on a dataset of real DIA data. Unfortunately our GAN model was not able to smooth the input. But it was able to somewhat learn a compression of the input space and output chromatograms with similar shape to the input without distorting quantifiable data.

2 Brief LC/MS Primer

LC/MS identifies and quantifies proteins by separating them by their hydrophobicity and mass. In general it is difficult to quantify whole proteins so we instead use peptides, which are sub-sequences of a protein. These peptides come off of a liquid column at a specific time which is dependent on the hydrophobicity of the peptide. This time is known as the peptide's Retention Time. Once the peptide comes off the column it is ionized and then becomes what is called a precursor ion. When precursor ions go into the mass spectrometer they are fil-



(a) Precursor Chromatogram

(b) Transition Chromatograms(Peak Group)

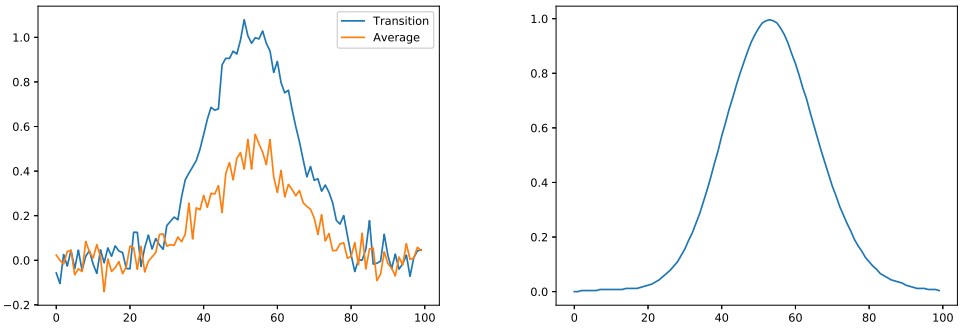
Figure 2: Here we show the difference between precursor and transition ion chromatograms. The precursor chromatogram contains a single peak, while the transition chromatogram shows a set of peaks coming from all the different transitions or fragment ions. This set of transition chromatograms coming from the same peptide is known as a peak group.

tered by the ratio of their mass to charge. A precursor ion's mass to charge along with its retention time are used as a two fold filtration process for unique identification and quantification of peptides.

At this point one could start quantifying the peptide, but in certain methods, like DIA, the precursor is then blasted it with an ion spray. This ion spray causes the precursor to fragment into two pieces. One of these pieces, known as a transition ion, becomes charged and may be detected by the machine. The fraction of the time the machine detects any possible fragment is relatively stable and known from empirical data. Because of this the relative ion intensity of each transition ion is useful for identification of the peptide. The relative intensity of each ion can be thought of as a unique key for a peptide.

3 Task Definition

The goal of this project is to train a GAN so that it can take a peak group and return the same peak group after being smoothed and denoised. The GAN smooths and denoises each transition separately. The network takes as input the chromatogram from a single transition as well as an average signal from all the other transitions coming from the same peak group. The reason this is done is that, as one can



(a) Input

(b) Output

Figure 3: Here we show the input and output data for the GAN network. The input composes of unique transition along with an average shape chromatogram coming from all other transitions in the peak group. The output is the smoothed peak group from the transition. (Note the above is not real input or output data as a result of training. It is only for demonstration)

see from figure 2b, each transition chromatogram has a similar shape to all the other chromatograms in the same peak group. So having an average of the other chromatograms is a clue into the true shape of the transition chromatogram we are trying to smooth.

The GAN takes the average chromatogram and the single transition chromatogram as input and then outputs a chromatogram of the single transition after it has been smoothed.

4 Algorithm Definition

4.1 Chromatogram Pre-processing

Before we can use the chromatogram data it has to be pre-processed. The time span of each peakgroup was not of equal length so all the chromatograms had to be up-sampled or down-sampled so that they were all a standard length of 100 intensity points per peak. As well the intensities were normalized to range from 0 to 1.

4.2 GAN

With this processed data, we trained a 1-Dimensional, denoising GAN. Because the data is effectively a 1-Dimension Image, we chose to adapt a popular Convolutional GAN architecture that has been successfully applied to images, known as DCGAN.

4.3 Noise Model

In order to remove the effect of the noise on the activation, we need to be able to generate examples of this noise. The best noise model would be the actual, underlying physical process. We do not have access to this, however, so we defined a noise model that is a Gaussian Process with an RBF Kernel with dimension 1. The advantage of our approach over just assuming some generative model for the peptide activation is that a noise model will potentially be much simpler because it depends on the data collection apparatus and not the structure of the peptide.

4.4 Architecture Decisions

In general, we used the architecture advice present in [4], with several alterations to fit our specific problem domain. Neither the discriminator, nor the generator has any type of pooling layer because the operation we want is smoothing, and there should be no "nonlocal-information" that would necessitate summary statistics from a pooling layer. Also, pooling layers have been shown to hurt convergence [4]. Batch normalization was used in both the Generator and the Discriminator, as were LeakyReLUs.

4.5 Generator Architecture

The Noise Model is sampled and added to a real example, and then fed into the generator. The Generator will then attempt to reconstruct the original image, minus the noise. The loss function on the generator is the squared difference between the denoised output image and the original image, and the standard $log(D)$, where D is the Discriminator's loss. Gradient Descent was achieved using the ADAM optimizer with $\epsilon = 0.01$. The number of parameters in the Generator

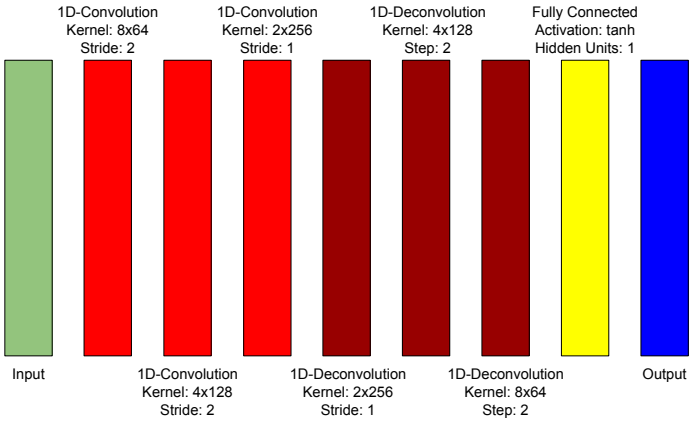


Figure 4: Generator Structure

decreases near the center of the network, in order to force it to generalize. We used dropout with $p = 0.5$ in both the training and testing of the Generator.

4.6 Discriminator Architecture

The inputs to the Discriminator are either images that come from the generator, or the real noiseless images, and it is the job of the Discriminator to distinguish between the two. Stochastic Gradient Descent with a learning rate of $\eta = 0.01$ was used to optimize the weights of the network.

5 Methodology

For training we used a high qualitiy SRM dataset [7] containing approximately 12,000 unique transition chromatograms. For our testing, we used DIA dataset [9] of 16 samples each containing 500 peakgroups.

For the test data, we only had the noisy raw DIA data. We don't know what the true shape of the peaks should look like once the noise was removed. For a single peak group the only way to know if we correctly smoothed the data without distorting it would be if we knew beforehand the true amount of protein that existed in the sample.

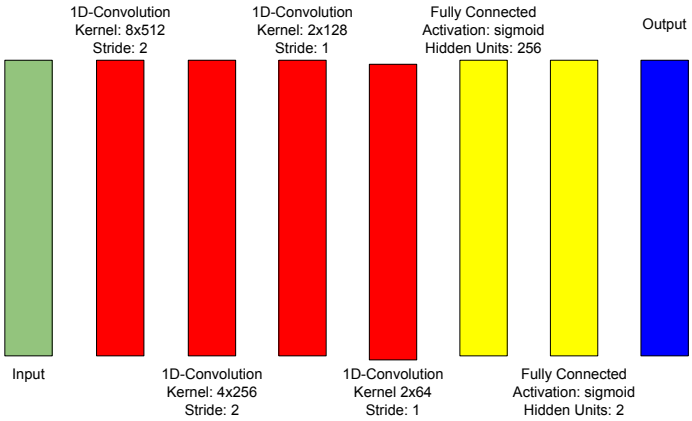


Figure 5: Discriminator Structure

Unfortunately we did not know that for the samples we were using so we had to use a different metric.

Instead we used relative ion intensity to measure the quality of our data. As mentioned earlier in the primer, the relative intensity of each fragment ion has a fairly stable value which we can use for identification. In the test set we have 16 different samples all containing the same peptides. If after smoothing the test data we found that the relative ion intensties for the same peptide across samples didn't match, then we would know that we had corrupted the data.

To calculate this metric we simply integrate over the curve of each ion's chromatogram in the same peak group. Combing these integrations gives a vector of chromatogram areas for each peptide in each sample. Then with these vectors we simply calculate the average cosine angle between all different samples for the same peptide. This average is our score, where 1 is the best and -1 is the worst.

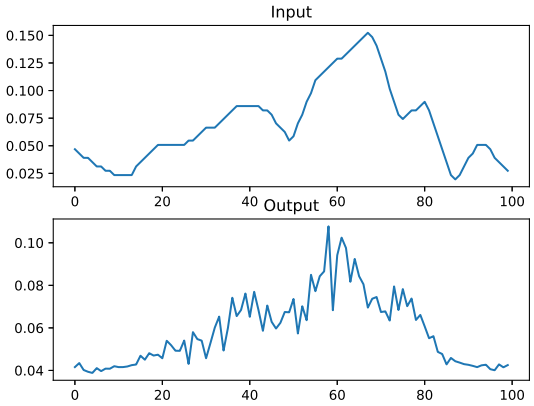
If after doing smoothing from GANS we see this score go significantly down from before smoothing, then we know that we are corrupting the data. But if it is close to 1, then we can feel confident that the GAN is smoothing the data correctly.

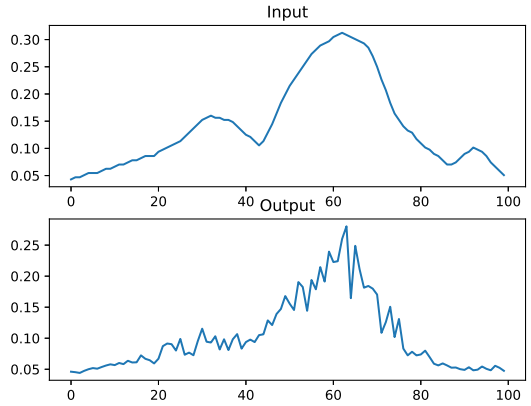
6 Results

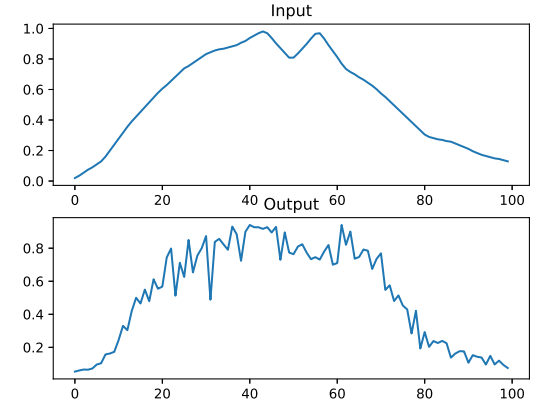
In the test set we achieved images that appeared to somewhat look like the original input. The data was also not corrupted. Both the

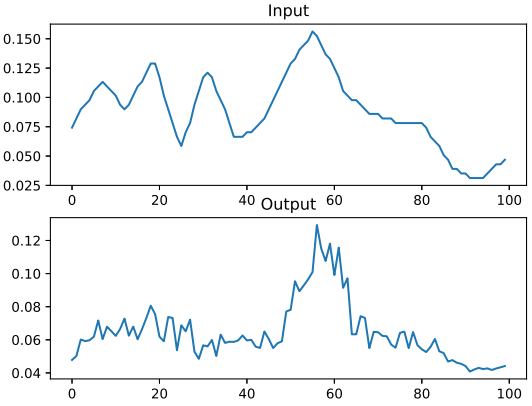
original DIA data and the predicted data got approximately the same dot product scores of ~ 0.98.

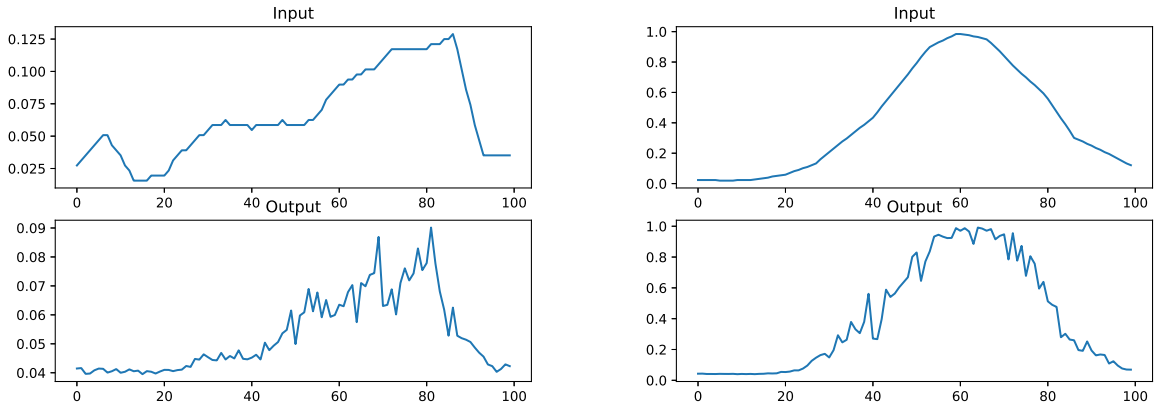
7 Test Predictions











8 Discussion

In previous efforts, groups have used what's known as a "Perceptural Loss Function" to help the Generator Generate images that "look like" the original image. This is done by minimizing the distance between the Generator's output and the high-level features derived by a state-of-the-art supervised image classification network. We wanted to do something similar, but lacked classification data on the activations. The lack of this part of the loss function shows. While the outputs come close to the original in Euclidian distance, they somehow don't look like the originals.

The convergence on the GAN was relatively speedy. We attribute this to the lack of ReLUs and sparse operations such as pooling.

The impact of the noise model can also be seen in the output. The generated activations look like the original with some sort of Gaussian noise removed, which is exactly what we were trying to get the networks to do.

9 Individual Contributions

9.1 Max Horowitz-Gelb

- Extracted data using Skyline [6] proteomics software.
- Formatted and ordered raw data
- Preprocessed chromatogram data using NumPy [10].

- Visualized Computational Graphs using TensorFlow [2]
- Created dot product evaluation code using NumPy.
- Generated loss summary graphs using Tensor Flow

9.2 Gabriel Ferns

- Created GANs net in TensorFlow
- Tuned training parameters

10 Related Work

Work has been done in the past to smooth chromatograms, but as far as we know, never with a neural network. [7] used standard statistical and signal processing methods to smooth and give better quantification of peak areas in SRM data. [11] removed noise as well as overlapping peaks using a gaussian mixture model. Another model [1] smoothed peaks by fitting them to a complicated piecewise function.

11 Future Work

The single biggest improvement that could be made here is a better noise model. There have been fairly successful efforts to quantify the biases of a particular machine in the past, but because of the novelty of the data collection technique, we did not have access to something similar.

More time could have been also spent tuning the hyper-parameters and running more epochs of the network. Converging to something decent in 3000 epochs is somewhat impressive.

Finally, it is unclear whether or not the Adversarial part of the Generator's loss function actually made it better at its job. Without the adversarial component, the generator is effectively an autoencoder, which could have produced the results we see. More investigation is needed.

12 Conclusion

As we've shown, our GAN is able to somewhat understand the distribution of peak shapes. Though it was not able to smooth the input

data it produced outputs from compressed information that somewhat resemble the shape of the input without distorting the quantifiable information. One of the main issues is that we did not have a noise model that was based in the physics of mass spectrometry. We just used a simple Gaussian process noise model. But the fact that we were able to somewhat recreate the original input is a good sign though. It suggests that the GAN network was able to somewhat do feature extraction and learned a generative model for all peaks. This could not only be useful for peak smoothing but also peak detection. The features learned in this model could be applied to a predictive task where we need to select out peaks from a long potentially hour long chromatogram. Though our model is not yet useable for peak smoothing on real data, it seems promising and with more work it could be a powerful tool for peak smoothing and potentially many other computational tasks involved in the field of mass spectrometry.

References

[1] Automated data review using ascent. *indigo bioAutomation*, 2015.

[2] Martn Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv*, 2016.

[3] Ludovic C. Gillet, Pedro Navarro, Stephen Tate, Hannes Rost, Nathalie Selevsek, Lukas Reiter, Ron Bonner, , and Ruedi Aebersold. Targeted data extraction of the ms/ms spectra generated by data-independent acquisition: A new concept for consistent and accurate proteome analysis. *Molecular and Cellular Proteomics*, 2012.

[4] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv*, 2015.

[5] Christian Ledig, Lucas Theis, Ferenc Huszar, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, and Wenzhe Shi. Photo-realistic single image super-resolution using a generative adversarial network. *arXiv*, 2016.

[6] Brendan MacLean, Daniela M. Tomazela, Nicholas Shulman, Matthew Chambers, Gregory L. Finney, Barbara Frewen, Randall Kern, David L. Tabb, Daniel C. Liebler, and Michael J. MacCoss. Skyline: an open source document editor for creating and analyzing targeted proteomics experiments. *Bioinformatics*, 2010.

[7] S. Nasso, S. Goetze, and L. Martens. Ariadne's thread: A robust software solution leading to automated absolute and relative quantification of srm data. *Journal of Proteome Research*, 2015.

[8] Paola Picotti and Ruedi Aebersold. Selected reaction monitoringbased proteomics: workflows, potential, pitfalls and future directions. *Nature Methods*, 2012.

[9] Hannes L Rost, Yansheng Liu, Giuseppe D'Agostino, Matteo Zanella, Pedro Navarro, George Rosenberger, Ben C Collins, Ludovic Gillet, Giuseppe Testa, Lars Malmstrom, and Ruedi Aebersold. Tric: an automated alignment strategy for reproducible protein quantification in targeted proteomics. *Nature Methods*, 2016.

[10] Stefan van der Walt, S. Chris Colbert, and Gael Varoquaux. The numpy array: A structure for efficient numerical computation. *Computing in Science and Engineering*, 2011.

[11] Tianwei Yu and Hesen Peng. Quantification and deconvolution of asymmetric lc-ms peaks using the bi-gaussian mixture model and statistical model selection. *BMC Bioinformatics*, 2010.

[12] He Zhang, Vishwanath Sindagi, and Vishal M. Patel. Image deraining using a conditional generative adversarial network. *arXiv*, 2017.