Generalized Regression with Conditional GANs

Deddy Jobson[1] and Eddy Hudson[2]

[1] Mercari Inc., Tokyo, Japan
deddy@mercari.com
[2] University of Texas Austin, Austin, USA
eddyhudson@utexas.edu

Abstract. Regression is typically treated as a curve-fitting process where the goal is to fit a prediction function to data. With the help of conditional generative adversarial networks, we propose to solve this age-old problem differently; we aim to learn a prediction function whose outputs, when paired with the corresponding inputs, are indistinguishable from feature-label pairs in the training dataset. We show that this approach to regression makes fewer assumptions on the distribution of the data we are fitting to and, therefore, has better representation capabilities. We draw parallels with generalized linear models in statistics and show how our proposal extends them to neural networks. We demonstrate the superiority of this new approach to standard regression with experiments on multiple synthetic and publicly available real-world datasets, finding encouraging results, especially with real-world heavy-tailed regression datasets. To make our work more reproducible, we release our source code[3].

Keywords: conditional generative adversarial networks, heavy-tailed distribution, regression, neural network

1 Introduction

Generative Adversarial Networks [5] (GANs) revolutionized how we generate realistic artificial images. Their success is owed to the fact that compared to other methods, they can more effectively represent intractable distributions such as images. This is because instead of using hand-designed closed-form loss functions to optimize the image generator, they use an adversarial discriminator to train the generator to produce realistic images. It stands to reason that this idea can potentially be applied to better represent probability distributions in general, not just that of images. This line of reasoning is exemplified by recent advances in reinforcement learning that capitalize on the generative adversarial framework to learn a behavior policy that is hard to specify [6] [8].

We study the application of conditional GANs (CGAN) on the problem of regression with tabular data. CGANs offer an alternative approach to the training of neural networks for regression tasks. Instead of directly regressing on the

[3] Link to our code: https://github.com/deddyjobson/regressGAN

target variable with a loss function like MSE, we instead train two models simultaneously, one to make predictions given only the input covariates and another to decide whether or not the predictions are distinguishable from the ground truth labels, again given the input covariates.

The following are our contributions:

- We show that using GANs for regression will require fewer assumptions on the distribution of data (Section 3).
- We perform experiments to demonstrate the superiority of regression with GANs against other regression methods (Section 4).
- We also empirically demonstrate that training GANs for the case of regression with tabular data requires fewer tricks than with image data (Section 6).

2 Background

2.1 Generative Adversarial Networks

GANs are an unsupervised method used to learn a generative model of a probability distribution. Conditional GANs [12] (CGANs) were developed as a GAN-based method to generate images conditioned on the input labels. Mirza et al. used CGANs in their seminal publication to generate realistic images of numbers from the MNIST dataset based on their ground truth labels.

2.2 Generalized Linear Models

In regression, the objective is to maximize the likelihood of the data fitting the model. The simplest form of regression is linear regression, which assumes that the distribution of the regression residuals follows a normal distribution. To account for violations of the assumption of normal residuals, statisticians use the link function to extend the representation ability of linear models. This results in generalized linear models [13].

For different industrial applications, specific link functions have been proposed. For example, to model customer revenue in e-commerce, assuming the residuals to follow heavy-tailed distributions like zero-inflated log-normal distribution [19], Tweedie distribution [20], etc. have been applied with success. While the above methods succeed in their respective domains, none of the likelihood functions used can be generally applied to all regression problems. To be able to do that effectively, we need a likelihood function that can itself adapt to new domains.

3 Our Method

We propose using CGANs to solve the regression problem. CGANs take a different approach to regression. Instead of maximizing the likelihood of the generated

predictions belonging to the true underlying distribution of the target variable, the goal is to generate predictions as indistinguishable from the ground truths as possible. The benefit of this approach is that, unlike with generalized linear models, we do not need to formulate the likelihood function explicitly; with enough data, the likelihood function, too, is learned by the CGAN. This has been proved by Goodfellow et al., and we use the notation and derivation from their paper [5] for our theoretical argument. More specifically, consider Proposition 1 and Equation 2 from their paper. The optimal discriminator for a fixed generator is

$$D^*_G(x) = \frac{p_{data}(x)}{p_{data}(x) + p_g(x)} \tag{1}$$

Also, from Equation 6 in Theorem 1, we see that given the optimal discriminator, the generator would optimize the Jensen Shannon divergence between the output distribution of the generator and that of the true distribution.

$$C(G) = -log(4) + 2JSD(p_{data}||p_g) \tag{2}$$

We see from the above equation that as long as the neural networks used in the GAN have sufficient representation capacity, we can directly optimize the Jensen Shannon Divergence between the prediction ($p_g(x)$) and true distributions ($p_{data}(x)$) without explicitly defining the true distribution.

In this paper, we use a CGAN to represent the distribution of the target variable (Y) conditioned on the dependent variables (x). Since we use the CGAN for regression, we shall refer to our approach as the RegressGAN method.

4 Experiments

In order to assess the capability of CGANs for regression, we compare their performance against baseline algorithms on several datasets.

4.1 Datasets

We perform experiments on three synthetic datasets and three real-world datasets. Two real-world datasets are publicly available, while the third dataset is proprietary.

We adopt the notation commonly used in statistics. We index each data point with i. The values of the independent/predictor variables for each data point i are represented by the vector x_i and the dependent, response, or target variable by y_i. Finally, we represent the predictions made by any model for each data point by $\hat{y_i}$ and the residuals (errors) of predictions by ϵ_i.

For all synthetic datasets, we take 100,000 random samples of data and split the data into the train (60%), validation (20%), and test (20%).

Synthetic-Normal We synthesize a linear dataset with Gaussian noise in the following way:

$$\boldsymbol{\beta} \sim MVN\left(0, \frac{1}{10}\boldsymbol{I}_{25}\right) \tag{3}$$

$$y_i = \mathbf{x_i}\boldsymbol{\beta} + \epsilon_i \tag{4}$$

$$\epsilon_i \sim \mathcal{N}(0, 1) \tag{5}$$

Synthetic-Heteroscedastic Following the work of Aggarwal et al. [1], we synthesize a dataset with Gaussian noise in the following way:

$$x_i \sim N\left(0, 1\right) \tag{6}$$

$$z_i \sim N\left(1, 1\right) \tag{7}$$

$$h_i = (0.001 + 0.5|x_i|) \times z_i \tag{8}$$

$$y_i = x_i + h_i \tag{9}$$

Synthetic-Classification We synthesize a dataset with a binary target variable in the following way:

$$\boldsymbol{\beta} \sim MVN\left(0, \frac{1}{2}\boldsymbol{I}_{25}\right) \tag{10}$$

$$p_i = Sigmoid(\mathbf{x_i}\boldsymbol{\beta}) \tag{11}$$

$$y_i \sim Bernoulli(p_i) \tag{12}$$

Synthetic-Tweedie The last synthetic dataset we use involves modeling the target variable with a distribution following the Tweedie distribution [3].

$$\boldsymbol{\beta} \sim MVN\left(0, \frac{1}{10}\boldsymbol{I}_{25}\right) \tag{13}$$

$$\mu_i = e^{\mathbf{x_i}\boldsymbol{\beta}} \tag{14}$$

$$y_i \sim Tweedie(\mu_i, 1.5, 1) \tag{15}$$

Car Insurance The first real dataset we use is the French Motor Third-Party Liability Claims dataset [14]. The dataset contains car insurance claims made over a year. We sampled 100k data points, from which we sampled 20k each for validation and test sets. We observe the data to have a heavy tail, as is usually the case with insurance claim data [11]. The dataset is, therefore, appropriate for testing the representative capability of regressGAN.

Health Insurance The next real-world dataset we use is the US Health Insurance dataset[4]. There are 1338 records in total. We again split 20% of the data

[4] https://www.kaggle.com/datasets/teertha/ushealthinsurancedataset

for each of the validation and test sets. Furthermore, we use one hot encoding on the categorical features to get 8 independent variables in total, which we will use to predict the same target as before: the amount of expense claimed.

E-commerce Lastly, we also perform experiments on the user logs provided by an online C2C marketplace where individuals can buy or sell items. We sample 100k users for our experiments. We use each user's historical data to create 12 features to forecast the future revenue generated by each user. Here, too, we get a zero-inflated heavy-tailed distribution for the target variable.

4.2 Models

We compare in total three models for our experiments:

- **RegressGAN**: This is our proposed method in which we use a conditional GAN for regression, which we described earlier.
- **FNN-MSE**: For our regression baseline, we use a feed-forward network (FNN) of the same architecture and other hyperparameters as the generator (without the noise input) with the Mean-Squared Error (MSE) loss function.
- **GP**: The final baseline we compare our method with is Gaussian Process Regression [18]. We use a Python implementation from scikit-learn [17] with the RBF kernel.

5 Results

To evaluate the performance of regressGAN, we measured the MAE of predictions on the test dataset since it is commonly used for evaluation when the response variable is heavy-tailed like in most of our experiments [10, 4].

$$MAE = \frac{1}{N}\sum_{i=1}^{N}\left|\frac{y_i - \hat{y}_i}{y_i}\right| \tag{16}$$

We first discuss the results obtained on the synthetic datasets in Table 1. Surprisingly, the regressGAN approach performs best in all but one dataset, even for the "Normal" dataset. We suspect this to be the case because while other models are prone to overfitting, the generator in regressGAN has the more difficult task of estimating the whole conditional distribution, making it more resilient to overfitting. For the heteroscedastic dataset, our results disagree with those of Aggarwal et al. in that the MSE model performs better. Perhaps it is because the underlying signal is very simple (being linear).

We next discuss the results obtained from the real-world datasets tabulated in Table 2. Unlike in the case of the synthetic datasets, the improvement in performance from regressGAN is considerable. For all datasets, we find RegressGAN to perform best among all algorithms. We suspect that the zero-inflation and high skewness of the target distributions made it difficult for the other algorithms to represent the complex distribution while respecting their strict assumptions efficiently.

Table 1. MAE of predictions on synthetic datasets (lower the better). Best results are highlighted in bold.

DATASET	FNN-MSE	GP	RegressGAN (ours)
Normal	**0.818**	0.844	**0.818**
Heteroscedastic	**0.329**	0.851	0.350
Classification	**0.262**	0.364	**0.262**
Tweedie	0.835	2.909	**0.805**

Table 2. MAE of predictions on real datasets. Best results are highlighted in bold.

Dataset	FNN-MSE	GP	RegressGAN (ours)
Car Insurance	0.358	0.420	**0.261**
Health Insurance	0.223	0.637	**0.178**
E-commerce	0.067	0.093	**0.059**

6 Ablation study on tricks used to speed up GAN training

We investigate tricks commonly used to help traditional GANs converge and see if we could do without them for RegressGAN. The authors who proposed GANs [5] made changes to the objective function of the GAN to help it converge. Specifically, rather than minimizing the log probability of correct predictions by the discriminator, they maximize the log probability of incorrect predictions by the discriminator. While this subtle trick was necessary in the context of image data to help GAN training, we them unnecessary with tabular data. Our experiments found no significant difference in convergence rate or in the final MAE by forgoing the training trick. This result is encouraging because they may make RegressGANs easier to deploy in production systems.

7 Related Work

Ours is not the first work to study the application of GANs for regression, though existing work is limited. Aggarwal et al. [1] considered the application of GANs to small regression problems where the distribution of the noise variable could not be explicitly modeled. We replicate their experiments on a synthetic heteroscedastic dataset that claims that feed-forward neural networks perform better. Oskarsson et al. [16] used CGANs for probabilistic regression. Rather than just learning a point estimate for each data point, the author used CGANs to estimate distributions. This work was mostly restricted to comparing different CGAN algorithms against each other.

Hudson et al. [7] employ CGANs to regress towards action distributions in reinforcement learning that have multiple modes. In their framework, the actions are generated conditioned on the state that the agent find itself in.

A key difference between the above works and ours is in the real-world datasets chosen for the experiments; all our datasets have heavy-tailed response variables, specifically where we see the superior performance of CGANs. To our knowledge, our work is the first to discover real-world evidence of the superiority of CGANs for heavy-tailed regression.

8 Limitations

In this paper, we demonstrate the superiority of the GAN formulation over other loss functions, such as the MSE loss function for neural networks. However, gradient-boosted machines (GBMs) are currently the state-of-the-art method for tabular data. While neural network architectures have been proposed to surpass them, none have gained widespread adoption [2] so far. Our method complements these attempts to create better neural networks for tabular data. When neural networks finally surpass GBMs over tabular data, our method can be applied to set neural networks further apart from traditional methods.

Furthermore, recent research with extreme value theory has suggested that GANs cannot generate heavy-tailed distributions [15, 9]. This runs counter to what we have empirically observed with insurance datasets. Perhaps while RegressGAN can represent a wide variety of conditional distributions better than neural regression models, there are some distributions that it cannot represent well.

9 Conclusion

In this paper, we highlight the representation capabilities of CGANs for regression tasks and show that they are superior to standard neural networks trained with the MSE loss function. They prove to be a flexible generalization of regression for neural networks in the same way generalized linear models are a flexible generalization of ordinary linear regression. While Oriol et al. [15] prove theoretically that GANs cannot represent some heavy-tailed distributions, we show that GANs can better represent heavy-tailed distributions in practice than traditional methods. Together with our discovery of the relative ease of training GANs for tabular data, our results suggest that CGANs have the potential to be used in even more novel applications in science and industry in the years to come, especially when handling data of unknown distribution.

References

1. Aggarwal, K., Kirchmeyer, M., Yadav, P., Keerthi, S.S., Gallinari, P.: Benchmarking Regression Methods: A comparison with CGAN (Feb 2020), arXiv:1905.12868 [cs, stat]

2. Borisov, V., Leemann, T., Seßler, K., Haug, J., Pawelczyk, M., Kasneci, G.: Deep Neural Networks and Tabular Data: A Survey. IEEE Transactions on Neural Networks and Learning Systems pp. 1–21 (2022). https://doi.org/10.1109/TNNLS.2022.3229161, conference Name: IEEE Transactions on Neural Networks and Learning Systems
3. Dunn, P.K., Smyth, G.K.: Series evaluation of Tweedie exponential dispersion model densities. Statistics and Computing **15**(4), 267–280 (Oct 2005). https://doi.org/10.1007/s11222-005-4070-y
4. Glady, N., Baesens, B., Croux, C.: A modified Pareto/NBD approach for predicting customer lifetime value. Expert Systems with Applications **36**(2, Part 1), 2062–2071 (Mar 2009). https://doi.org/10.1016/j.eswa.2007.12.049
5. Goodfellow, I.J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative Adversarial Networks (Jun 2014). https://doi.org/10.48550/arXiv.1406.2661, arXiv:1406.2661 [cs, stat]
6. Ho, J., Ermon, S.: Generative adversarial imitation learning. Advances in neural information processing systems **29** (2016)
7. Hudson, E., Durugkar, I., Warnell, G., Stone, P.: Abc: Adversarial behavioral cloning for offline mode-seeking imitation learning. arXiv preprint arXiv:2211.04005 (2022)
8. Hudson, E., Warnell, G., Torabi, F., Stone, P.: Skeletal feature compensation for imitation learning with embodiment mismatch. In: 2022 International Conference on Robotics and Automation (ICRA). pp. 2482–2488. IEEE (2022)
9. Huster, T., Cohen, J.E.J., Lin, Z., Chan, K., Kamhoua, C., Leslie, N., Chiang, C.Y.J., Sekar, V.: Pareto GAN: Extending the Representational Power of GANs to Heavy-Tailed Distributions (Jan 2021). https://doi.org/10.48550/arXiv.2101.09113, arXiv:2101.09113 [cs]
10. Jasek, P., Vrana, L., Sperkova, L., Smutny, Z., Kobulsky, M.: Comparative analysis of selected probabilistic customer lifetime value models in online shopping. Journal of Business Economics and Management **20**(3), 398–423 (Apr 2019). https://doi.org/10.3846/jbem.2019.9597, number: 3
11. Mikosch, T.: Heavy-tailed modelling in insurance. Communications in Statistics. Stochastic Models **13**(4), 799–815 (Jan 1997). https://doi.org/10.1080/15326349708807452, publisher: Taylor & Francis _eprint: https://doi.org/10.1080/15326349708807452
12. Mirza, M., Osindero, S.: Conditional Generative Adversarial Nets (Nov 2014), arXiv:1411.1784 [cs, stat]
13. Nelder, J.A., Wedderburn, R.W.M.: Generalized Linear Models. Journal of the Royal Statistical Society: Series A (General) **135**(3), 370–384 (1972). https://doi.org/10.2307/2344614, _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.2307/2344614
14. Noll, A., Salzmann, R., Wuthrich, M.V.: Case Study: French Motor Third-Party Liability Claims (Mar 2020). https://doi.org/10.2139/ssrn.3164764
15. Oriol, B., Miot, A.: On some theoretical limitations of Generative Adversarial Networks (Oct 2021). https://doi.org/10.48550/arXiv.2110.10915, arXiv:2110.10915 [cs]

16. Oskarsson, J.: Probabilistic Regression using Conditional Generative Adversarial Networks (2020)
17. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research **12**(85), 2825–2830 (2011)
18. Wang, J.: An Intuitive Tutorial to Gaussian Processes Regression (Apr 2022), arXiv:2009.10862 [cs, stat]
19. Wang, X., Liu, T., Miao, J.: A Deep Probabilistic Model for Customer Lifetime Value Prediction (Dec 2019). https://doi.org/10.48550/arXiv.1912.07753, arXiv:1912.07753 [stat]
20. Yang, J., Li, Y., Jobson, D.: Personalized Promotion Decision Making Based on Direct and Enduring Effect Predictions (Jul 2022). https://doi.org/10.48550/arXiv.2207.14798, arXiv:2207.14798 [cs]