A Comparative Study of Non-Normal Distributions in Continuous Dropout

By
Alexander Guyer

A THESIS

submitted to

Oregon State University

Honors College

in partial fulfillment of
the requirements for the
degree of

Honors Baccalaureate of Science in Computer Science
(Honors Scholar)

Presented May 23, 2020
Commencement June 2020

AN ABSTRACT OF THE THESIS OF

Alexander Guyer for the degree of Honors Baccalaureate of Science in Computer Science presented on May 23, 2020. Title: A Comparative Study of Non-Normal Distributions in Continuous Dropout

Abstract approved:

William Smart

Recent studies have shown that novel continuous dropout methods can be viewed as a Bayesian interpretation of model parameters, though most such studies have shown results using normal distributions. As the posterior distributions over neural network nodes and parameters are intractable, given that they are a result of artificial construction to improve model performance rather than a result of observation, there is no justification in assuming that they are necessarily normal. In this paper, a unimodal and symmetric distribution called the generalized normal distribution, sometimes referred to as the exponential power distribution, is instantiated with various shape and scale parameter configurations. These instantiated distributions are tested as nodal representations in multilayer perceptrons trained against the MNIST and MNIST Fashion datasets. Results conclude that the shape parameter of a generalized normal distribution has a statistically significant effect on the performance of the multilayer perceptron in continuous dropout against MNIST. Results also suggest, though not conclusively, that a Gaussian distribution is not necessarily optimal in continuous dropout against MNIST.

Key Words: neural network, dropout, overfitting

Corresponding e-mail address: guyera@oregonstate.edu

A Comparative Study of Non-Normal Distributions in Continuous Dropout

By
Alexander Guyer

A THESIS

submitted to

Oregon State University

Honors College

in partial fulfillment of
the requirements for the
degree of

Honors Baccalaureate of Science in Computer Science
(Honors Scholar)

Presented May 23, 2020
Commencement June 2020

Honors Baccalaureate of Science in Computer Science project of Alexander Guyer presented on May 23, 2020

APPROVED:

William Smart, Mentor, representing Mechanical Industrial and Manufacturing Engineering

Fuxin Li, Committee Member, representing Electrical Engineering and Computer Science

Lan Xue, Committee Member, representing Statistics

Toni Doolen, Dean, Oregon State University Honors College

Alexander Guyer, Author

Contents

A Comparative Study of Non-Normal Distributions in Continuous Dropout

Alexander Guyer
Electrical Engineering and Computer Science
Oregon State University
Corvallis, OR, United States
guyera@oregonstate.edu

***Abstract*—Recent studies have shown that novel continuous dropout methods can be viewed as a Bayesian interpretation of model parameters, though most such studies have shown results using normal distributions. As the posterior distributions over neural network nodes and parameters are intractable, given that they are a result of artificial construction to improve model performance rather than a result of observation, there is no justification in assuming that they are necessarily normal. In this paper, a unimodal and symmetric distribution called the generalized normal distribution, sometimes referred to as the exponential power distribution, is instantiated with various shape and scale parameter configurations. These instantiated distributions are tested as nodal representations in multilayer perceptrons trained against the MNIST and MNIST Fashion datasets. Results conclude that the shape parameter of a generalized normal distribution has a statistically significant effect on the performance of the multilayer perceptron in continuous dropout against MNIST. Results also suggest, though not conclusively, that a Gaussian distribution is not necessarily optimal in continuous dropout against MNIST.**

***Index Terms*—neural network, dropout, overfitting**

I. Introduction

Neural networks are a class of nonlinear mathematical models in which nodes are connected via directed weighted edges, wherein the weights, also referred to as the model's **parameters**, serve as coefficients for the input node's value, and those parameters are incrementally adjusted to minimize the model's error given a set of training data. This training process is generally referred to as hill climbing, or gradient descent. Along with many other fundamental practices of machine learning, gradient descent was discussed in depth in Marvin Minsky's seminal paper, "Steps Toward Artificial Intelligence," in which he explained how gradients of the model's error with respect to each parameter can be computed to determine the direction, and optionally inform about the magnitude, by which to adjust the respective parameter [1]. By incrementing a parameter by an amount proportional to the negative of its computed gradient, the model's error will decrease, so long as the parameter is not over-adjusted.

A. Activation Functions

In effect, if a line of best fit is to be considered a linear statistical model used to correlate variables and provide output predictions, a neural network can be considered a complex non-linear model which exists to serve the same purpose. The non-linearity is introduced by non-linear **activation functions**. Nodes are organized in layers, and, in the most traditional case, nodes from each layer are connected to nodes in the following layer by individual edges weighted by their parameters. For a given node in the following layer, the summation of its inputs, each of which is the product between the corresponding parameter and input node, is provided as the input for the activation function. Traditionally, nodes within the same layer share the same activation function, though activation functions may and generally do differ across layers.

The most traditional activation function is the sigmoidal curve with a range of $(0, 1)$. It has been proven that the superposition of an arbitrary finite number of sigmoidal activations, as well as other activation functions under mild assumptions, can approximate any continuous function within a compact subset of $\mathbb{R}^n$ to an arbitrary degree of accuracy [2]. This theorem is fundamental to neural network theory, and it is known as the universal approximation theorem. The key constraint to this theorem is that the inputs must be contained within a compact subset. Ideally, a neural network trained on a given set of training data will correctly approximate the unknown non-linear map given the same training data as input. However, it is only useful as a prediction model if it can also interpolate or extrapolate. Given a complex non-monotonic function, a neural network may have a difficult time extrapolating. However, given sufficient training data for a task with bounded inputs, such as RGB values of an image in the task of object recognition, all inputs thereafter may be some case of interpolation. If the function is not highly sporadic, interpolation may be fairly successful, allowing the neural network to correctly predict output values corresponding to previously unseen inputs.

Recently, novel activation functions have provided an increase in neural network performance for a variety of tasks, including both classification and regression. The most popular of such activation functions is the rectified linear unit, or RELU activation function. It is defined by its positive domain, in which its range is linear and monotonically increasing, while its negative domain is often mapped to zero or linearly scaled by a relatively small coefficient (in the case of leaky RELU); it has been empirically shown that rectified linear units often yield greater predicition performance than hyperbolic tangents and sigmoid activation functions [3].

B. Local Extrema Convergence

When incrementally adjusting parameters in gradient descent, one will not always discover the global minimum of the model's error. Rather, it is possible that gradient descent will cause the model to descend into a local peak or trough wherein the incremental adjustment of any parameter will increase local error even if such a sacrifice is ultimately necessary to locate a more optimal extremum [1]. The occurrences of such problems are difficult to detect, leading to two practical solutions:

1) Increase the amount of training data so that a single local minimum is less likely to be shared across all input vectors
2) Increase the complexity of the network by superimposing a greater number of activation functions so that a single local minimum is less likely to be shared across all parameters

However, **Solution 1** is not always feasible. Often, training data is collected once or pulled from a public dataset. In such cases, one may be inclined to choose **Solution 2** and increase the number of nodes in the network so as to increase the number of activation function instances which are superimposed. However, this introduces a model complexity which is unnecessary for the given task environment. There are many other reasons which would also incline one to increase model complexity, such as convergence toward an otherwise insufficient absolute extrema or in the occasional case wherein training dataset accuracy is prioritized over the model's ability to interpolate and extrapolate (often referred to as **generalization**). The latter mentioned reason is rarely intentional, but more often an undesired consequence of increased model complexity.

C. Overfitting

Besides increasing training time, testing time, and memory requirements, increasing the complexity of a model by superimposing a greater number of activation functions is likely to introduce a problem known as **overfitting**. Formally, overfitting is defined as the case when the model's accuracy within the training input space is greater than that of the the complimentary input space, thus it can predict outputs within the training set much more effectively than it can predict outputs within the remainder of the problem space.

At least to some degree, overfitting is unavoidable; the universal approximation theorem suggests that it is theoretically possible to construct a simple multilayer perceptron which can achieve perfect accuracy over an arbitrary training set by approximating the corresponding function, but there is no such theorem which suggests that this approximated function will sustain its accuracy when applied to interpolated or extrapolated problems in the same problem space. This is because there are infinitely many functions which can achieve perfect accuracy within an arbitrary finite problem space, but only one which can achieve perfect accuracy within a problem space which is infinite or perceived as such. This concept can also be understood with a VC-dimension analysis. As the number of parameters increases along with the number of activation functions, the model's Vapnik-Chervonenkis dimension increases as well, meaning that it is capable of approximating a larger range of functions via superposition of its activation functions [4]. In other words, given a model that is perfectly capable of approximating the desired function, adding parameters will introduce alternative functions which the model is capable of approximating. If one of these newly introduced functions also correctly maps the training input space to the output space, then the

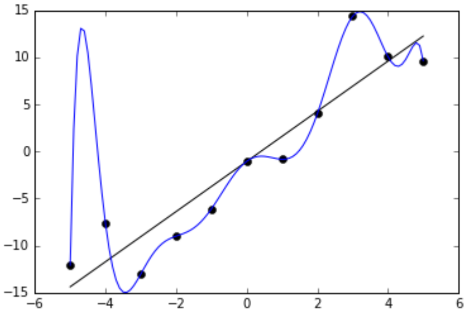


Fig. 1. A common depiction of overfitting resulting from unnecessary model complexity and consequential high-magnitude gradients

model may be equally likely to converge on this function as the original desired function. Given that the two functions are different, albeit equal within the testing input space, this newly introduced function will be less accurate within the training input space's complement. If the model converges on this function, overfitting will occur. Thus, increasing the complexity of a neural network may reduce the likelihood of converging on a local optimum, but it also increases the likelihood of introducing additional overfitting.

The discovery of this issue led to efforts to decrease overfitting in neural networks through techniques classified as regularization. One of the first widely used regularization methods is known as **L2 regularization**. In realistic task environments, ideal gradients tend to have relatively small magnitudes. As an overfitting model may correctly predict the output space given the training set, though falters when attempting to generalize, such a model is likely to have too extreme of output gradients with respect to the input space (i.e. a small change in the model's inputs results in a significant change in the model's prediction, contrary to the aforementioned description of realistic task environments). This is demonstrated in **Figure 1**.

As these gradients are solely determined by the magnitude of the model's parameters, the model's loss function can be modified to encourage convergence toward smaller, though still correct parameters. In L2 regularization, this is done by adding the square of the Euclidean norm of the flattened vector of all model parameters, multiplied by a constant, to the model's loss; thus, the larger in magnitude a given parameter is, the more it contributes to the model's error. As gradient descent aims to decrease the error described by the loss function, the model will tend toward smaller, though ideally still well-predicting parameters. Adjusting the supplied constant will directly adjust the significance of the squared Euclidean norm in the model's loss, thus adjusting the priority of regularization compared to prediction accuracy. Other similar regularization methods, such as L1 regularization, use different exponents when summing the model's parameters, effectively adjusting the sparseness of the solution set.

While L2 regularization and similar methods are highly effective, they assume that overfitting is merely a global problem with respect to the model's parameter set. However, it may be such that the ideal parameter set includes some parameters of proportionally high magnitudes when compared to other parameters. As such, a relatively high magnitude does not necessarily imply that the parameter is overfitting. However, methods such as L2 regularization tend to encourage all parameters to be relatively small, even if certain high-magnitude parameters are greatly contributing to the model's accuracy. The significance of this effect increases as the regularization exponent increases, as higher exponents yield more significant losses with relatively high-magnitude parameters.

In an effort to avoid such a divergence from an ideal solution associated with L2 and similar regularization methods, **early stopping** was developed, which recognizes that overfitting is a local issue and should not be treated equivalently across the entire model. It has been shown that large, complex neural networks pass through similar stages to those passed through by smaller networks when training on the same sample space. As such, even though the final result of the larger model's convergence may result in greater overfitting than that of the smaller model, the larger model will, at some point, pass through a stage which is very similar to the optimal convergence of the smaller model. As such, by terminating training of the larger model before its final convergence, it is possible to stop training at a point where the larger model is equally or more accurate than the terminally converged smaller model, avoiding the introduction of additional overfitting [5]. As this method simply terminates training before any overfitting occurs, it does not force those select parameters to diverge from their ideally higher-magnitude values, and thus does not treat regularization is a global mechanism to be applied equivalently to all parameters.

However, determining when to terminate training is difficult, as models tend to both increase and decrease in accuracy fairly sporadically during the training process. And while it may be feasible to cache optimal model states or use alternative caching methods to retrieve previous optimal model states, there is no guarantee that the stages passed by the larger model are identical to those of the smaller model. As such, the final result may still be less accurate than the terminally converged smaller model. This has led others to develop alternative methods to reduce overfitting in neural networks by artificially decreasing its complexity during training time while restoring its complexity and therefore its expressive capabilities during testing time.

D. Neuron Dropout

The most popular method of artificially altering a single network's complexity to reduce overfitting is known as **Bernoulli dropout** (traditionally referred to simply as "dropout") [6]. Bernoulli dropout was described in one of Nitish Srivastava's papers as a practical alternative to averaging several large models wherein random nodes are selectively omitted from a single super-model in each training case to achieve random sub-models. During testing time, when all of the nodes are utilized, the model's parameters are expected to be overestimated to compensate for the reduced complexity during training time. Thus, nodal outputs are multiplied by a restoring coefficient during testing time when all nodes are acknowledged to ensure that the actual model output during testing time converges in probability to the model's expected output during training time [7].

However, in the same paper, it was shown that the noise applied to a given node or alternative model parameter can be drawn from a continuous distribution, such as a Gaussian distribution, rather than from a discrete Bernoulli distribution. Specifically, Srivastava showed that a Gaussian distribution with mean 1 and a hyperparameterized variance σ, used to generate multiplicative noise to scale model parameters during training time, can perform as well as or better than Bernoulli dropout; it was also noted that this concept of **Gaussian dropout** can be generalized to any continuous distribution. This generalized method will be referred to as **continuous dropout**.

More recently, continuous dropout has been reinterpreted as a variational method to describe model parameters as probability distributions rather than point estimates. At the same time, it has been shown that the parameters used to represent continuous dropout distributions can be learned as well during training time by minimizing the KL-divergence between the dropout distribution and the intractable posterior distribution over the model parameters, done in practice by maximizing the variational lower bound of the marginal likelihood of the data observed [8]. This method is referred to as **variational dropout**. However, the minimum KL-divergence attainable through such methods is dependent entirely on the shape of the chosen continuous distribution and how it compares to the shape of the posterior distribution over the model parameters.

II. METHODS

As the posterior distribution over a model parameter is intractable, this paper empirically evaluates potential shapes of the average posterior distribution over an arbitrary node in a multilayer perceptron. Given that the average distribution's shape may be dependent on factors such as the model's architecture and task, this paper focuses solely on a multilayer perceptron and low-resolution image classification, though these methods can be extended to any architecture or task.

A. Generalization of Normal PDF

While a normal distribution has been used in both traditional continuous dropout and variational dropout, there is no evidence suggesting that the true shape of the intractable nodal or parameter distributions are necessarily normal. However, some success has been seen using normal distributions to approximate said intractable distributions. As such, in order to derive alternative distributions, the normal distribution was generalized to include a shape parameter, resulting in the following probability density function (sometimes referred to as an exponential power distribution):

$$f(y) = ce^{-|\frac{x}{\alpha}|^{\beta}}$$

In the given equation, c is an integral-normalizing constant within the window of interest to ensure a proper probability density function. This generalized normal distribution has the following properties:

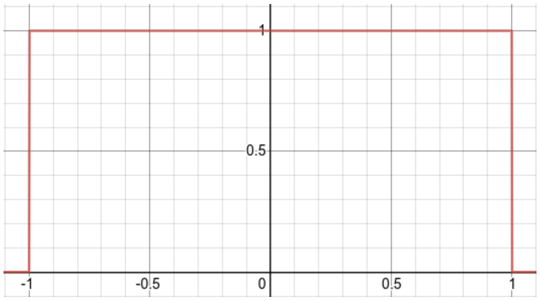


Fig. 2. Generalized normal PDF with $\beta = 10^{10}$, $\alpha = 1$

1) As $\beta \to \infty$, the distribution converges to a uniform distribution between $-\alpha$ and α
2) As $\beta \to 2$, the distribution converges to a normal distribution with mean $\mu = 0$ and variance $\sigma^2 = \frac{\alpha^2}{2}$

Property 1 is illustrated in **Figure 2**. It can also be derived from the following:

$$\begin{aligned}
\lim_{\beta \to \infty} ce^{-|\frac{x}{\alpha}|^{\beta}} &= \begin{cases} c & , |\frac{x}{\alpha}| < 1 \\ 0 & , |\frac{x}{\alpha}| > 1 \\ \frac{c}{e} & , |\frac{x}{\alpha}| = 1 \end{cases} \\
&= \begin{cases} \frac{1}{2\alpha} & , |x| < \alpha \\ 0 & , |x| > \alpha \\ \frac{1}{2e\alpha} & , |x| = \alpha \end{cases} \\
&= \begin{cases} \frac{1}{2\alpha} & , -\alpha < x < \alpha \\ 0 & , x < -\alpha, x > \alpha \\ \frac{1}{2e\alpha} & , x = \pm\alpha \end{cases}
\end{aligned}$$

The discrete case $x = \pm\alpha$ can effectively be ignored as the distribution is continuous, so the absolute probability of a discrete event is treated as zero. This yields the following result:

$$\lim_{\beta \to \infty} ce^{-|\frac{x}{\alpha}|^{\beta}} = \begin{cases} \frac{1}{2\alpha} & , -\alpha < x < \alpha \\ 0 & , otherwise \end{cases}$$

This can be viewed as the probability density function of a uniform distribution with bounds $-\alpha$ and α.

Property 2 is illustrated in **Figure 3**. It can also be derived from the following:

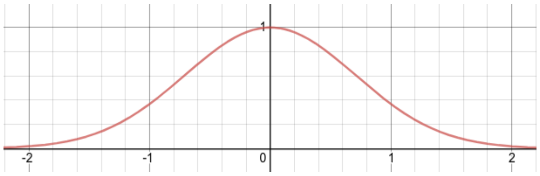


Fig. 3. Generalized normal PDF with $\beta = 2$, $\alpha = 1$

$$\begin{aligned}
f(y) &= ce^{-|\frac{x}{\alpha}|^2} \\
&= ce^{-(\frac{x}{\alpha})^2} \\
&= ce^{-\frac{1}{2}\frac{(x-0)^2}{\alpha^2/2}} \\
&= \frac{1}{\sqrt{2\pi\alpha^2/2}} e^{-\frac{1}{2}\frac{(x-0)^2}{\alpha^2/2}}
\end{aligned}$$

This can be viewed as the probability density function of a normal distribution with mean $\mu = 0$ and variance $\sigma^2 = \frac{\alpha^2}{2}$.

Given these two properties, the generalized normal PDF can almost be viewed as a generalization on continuous probability distributions which are unimodal and symmetric. However, this is not perfectly accurate, as certain uncommon unimodal symmetric probability density functions are not an instance of this class, such as the following:

$$f(y) = \begin{cases} 1 - |y| & , -1 \leq y \leq 1 \\ 0 & , otherwise \end{cases}$$

B. Probability Integral Transform

Simple hashing algorithms make pseudorandom uniform sampling trivial. However, sampling from an unconventional distribution requires transformations. The goal is to randomly sample from a uniform distribution, and then to somehow shift the sampled variable so that, upon repetition, the shifted distribution matches the desired alternative distribution.

Two of the defining properties of a distribution are its probability density function and its integral (cumulative distribution function). The area under a given interval of the probability density function (and so the corresponding range in the cumulative distribution function) marks the probability of observing an event in that interval. As such, $X\%$ of observed events will come from the interval bounding the first $X\%$ of the probability density function's area. In a uniform distribution, the relative probability of observing each event is constant.

Thus, in a standard uniform distribution (bounded between 0 and 1), the probability of observing an event less than or equal to Y is exactly Y, within the bounds [0, 1].

The intuition then follows: if a random variable sampled from a uniform distribution is observed to be Y, one can find the upper bound B of the lower interval within the alternative distribution's probability density function whose area also sums to Y; the probability of observing an event less than or equal to B within the alternative distribution will be equal to Y, just as is the probability of observing an event less than or equal to Y within the standard uniform distribution. This transformation is simply the inverse of the probability integral transformation, which yields a standard uniform distribution from an alternative distribution through integration [9].

The generalization of the normal PDF is not integrable by elementary means. It may be approximated by an infinite series which is trivially integrable such as a taylor series, but the result often diverges quickly from the true distribution without sufficient terms. As such, in order to integrate the generalized normal PDF for use in the probability integral transform, the trapezoidal integral approximation was used. The integral need only be approximated once to compute a table from which one can sample cumulative probabilities for an arbitrary probability integral transformation.

As the integral is represented using a table of trapezoidal area sums, a binary search is used to discover the closest approximation to the desired cumulative area Y and its corresponding upper bound B.

Lastly, the probability integral transformation traditionally transforms between any continuous probability distribution and a standard uniform distribution. However, if the original distribution is scaled so that its integral is non-normal, and thus so that it is no longer a valid probability distribution, the probability integral transformation can still be applied to yield a uniform distribution bounded in [0, C] where C is equal to the definite integral between $(-\infty, \infty)$ of the scaled probability density function. The proof is trivial: the scaling constant applied to the probability density function scales the cumulative distribution function by the same amount. Thus, the definite integral between $(-\infty, \infty)$ of the scaled probability density function, C, is equal to the scaling constant. The uniform random variable resulting from the probability integral transformation, then, will also be scaled by C. Scaling a uniformly distributed variable by a constant is equivalent to scaling its bounds by the same constant. Therefore, the result is a uniform distribution bounded in [0, C].

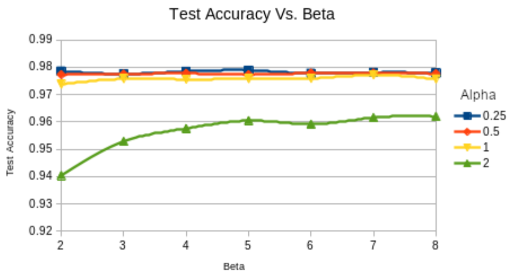


Fig. 4. A graph depicting the MNIST test accuracy versus β with multiple data series distinguished by α

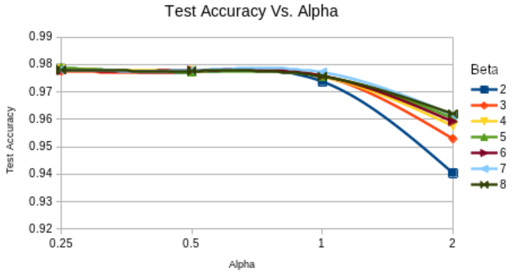


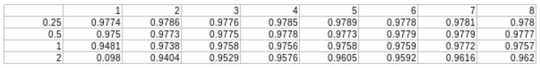
Fig. 5. A graph depicting the MNIST test accuracy versus α with multiple data series distinguished by β

Because of this property of the probability integral transform, the normalizing constant of the generalized normal PDF can effectively be ignored, as the function need only be transformed into a uniform distribution for sampling; this is possible without the generalized normal PDF having a normalized cumulative area.

III. Results

Various multilayer perceptrons were trained against the MNIST and MNIST Fashion datasets for a comparative study. Each multilayer perceptron differed only in hyperparameters which described the probability distribution of its nodes, α and β. Each MLP was equipped with two hidden layers, each with 512 nodes. Cross entropy was the loss function of choice, given the classification task at hand. Each MLP was trained for 20 epochs.

With the given MLP architecture against MNIST, it was found that, given $\beta = 2$, the most accurate



	1	2	3	4	5	6	7	8
0.25	0.9774	0.9786	0.9776	0.9785	0.9789	0.9778	0.9781	0.978
0.5	0.975	0.9773	0.9775	0.9778	0.9773	0.9779	0.9779	0.9777
1	0.9481	0.9738	0.9758	0.9756	0.9758	0.9759	0.9772	0.9757
2	0.098	0.9404	0.9529	0.9576	0.9605	0.9592	0.9616	0.962

Fig. 6. A table depicting the MNIST test accuracy versus α (row-grouped) and β (column-grouped)

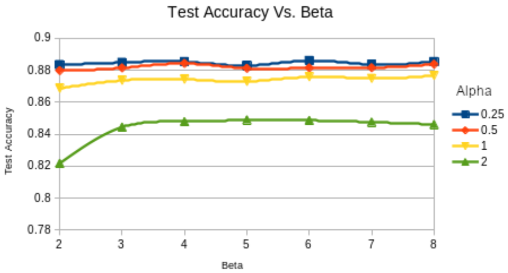


Fig. 7. A graph depicting the MNIST Fashion test accuracy versus β with multiple data series distinguished by α

result was achieved with $\alpha = 0.25$ and a test accuracy of 0.9786. This corresponds to a normal distribution with variance $\sigma^2 = \frac{1}{32}$. However, of greater importance is the finding that the most accurate result of the MNIST grid search occurred with $\beta = 5$ and $\alpha = 0.25$ and with a test accuracy of 0.9789. The table of results are depicted in **Figure 6**. Graphs depicting performance versus the generalized normal distribution's parameters are shown in **Figure 4 and Figure 5**.

With the given MLP architecture against MNIST Fashion, it was found that, given $\beta = 2$, the most accurate result was achieved with $\alpha = 0.25$ and a test accuracy of 0.8834. This also corresponds to a normal distribution with variance $\sigma^2 = \frac{1}{32}$. Again, of greater importance is the finding that the most accurate result of the MNIST fashion grid search occurred with $\beta = 6$ and $\alpha = 0.25$, with a test accuracy of 0.886. The table of results are depicted in **Figure 9**. Graphs depicting performance versus the generalized normal distribution's parameters are shown in **Figure 7 and Figure 8**.

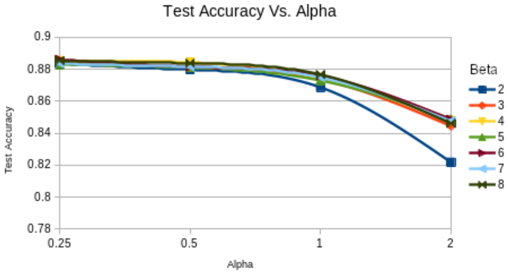
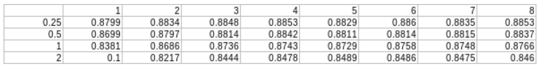


Fig. 8. A graph depicting the MNIST Fashion test accuracy versus α with multiple data series distinguished by β



	1	2	3	4	5	6	7	8
0.25	0.8799	0.8834	0.8848	0.8853	0.8829	0.886	0.8835	0.8853
0.5	0.8699	0.8797	0.8814	0.8842	0.8811	0.8814	0.8815	0.8837
1	0.8381	0.8686	0.8736	0.8743	0.8729	0.8758	0.8748	0.8766
2	0.1	0.8217	0.8444	0.8478	0.8489	0.8486	0.8475	0.846

Fig. 9. A table depicting the MNIST Fashion test accuracy versus α (row-grouped) and β (column-grouped)

As suspected, increasing the α hyperparameter tended to decrease the final accuracy after a certain point, depending on the value of β. However, prior to that point being exceeded, a higher α value tended to increase the final accuracy by reducing overfitting without significantly increasing the model's convergence time. Thus, until a threshold is exceeded, a higher α hyperparameter tends to result in a greater test accuracy and a lower training set accuracy. With a sufficiently high α value, however, the model is unable to converge quickly due to the high variance in nodal outputs, or sometimes unable to converge at all (as in both cases with $\alpha = 2, \beta = 1$) without reducing the training rate to compensate.

Variance, which is highly affected by the scale parameter α, is already understood to be an important parameter when implementing continuous dropout. However, in order to test the significance of the shape parameter β, K-fold cross validation was used with $K = 10$. Particularly, the two parameter configurations with equal variances that resulted in the widest range of of test accuracies in the MNIST dataset, being $(\beta = 1, \alpha = 1)$ and $(\beta = 7, \alpha = 1)$, were both tested using K-fold cross validation and the results were compared against one another using a two-tail matched-pairs t test (wherein each pair of samples was trained and tested on a fixed partition of the training data). The result of the t-test indicated a T score of 52.1938 and, with nine degrees of freedom, a p-value less than 0.00001.

Similarly, the optimally performing normal distribution of MNIST $(\beta = 2, \alpha = 0.25)$ was compared against the optimally performing generalized normal distribution of MNIST given the same variance $(\beta = 5, \alpha = 0.25)$, also using K-fold cross validation with $k = 10$ and a one-tail matched-pairs t test. The result of the t-test indicated a T score of 1.7434 and, with nine degrees of freedom, a p-value of 0.057626.

IV. Significance

In the field of machine learning, it is common to perform K-fold cross validation with K=10 in the way mentioned and compare results using a significance level of 0.05. This is the manner in which the results will be analyzed and discussed.

The first of the two p-values can be interpreted as the probability of observing such a large difference in performance in continuous dropout between two generalized normal distributions varying only in their shape parameters (holding scale parameters equal), assuming that the shape parameter has no effect on the performance of the multilayer perceptron. With a significance level of 0.05 and a p-value less than 0.00001, it can be stated that the results support the statement that performance in continuous dropout against MNIST can be improved by altering the shape parameter of the generalized normal distribution.

The second of the two p-values can be interpreted as the probability of observing such a large increase in performance in continuous dropout between a normal distribution and a generalized normal distribution with the same scale parameter but a different shape parameter, assuming that the shape parameter has no effect on the performance of the multilayer perceptron. With a significance level of 0.05 and a p-value of 0.057626, the data is not quite conclusive that a Gaussian distribution can necessarily be outcompeted in performance against MNIST by an alternative parameter configuration of a generalized normal distribution, holding the scale parameters (α) equal.

Thus, it can be stated that the shape parameter of a generalized normal distribution does have a statistically significant effect on the performance of continuous dropout against the MNIST dataset when approximating nodal posterior distributions.

Secondly, given a fairly low p-value in the second test, the data does seem to support the notion that a Gaussian distribution is not necessarily optimal in continuous dropout against the MNIST dataset. However, the data is not sufficiently significant to make any conclusions in this regard.

V. Further Research Opportunities

Given a low p-value in the second test, though not sufficiently low to permit any conclusions, the experiments could be performed with a higher sample size (thus requiring a higher K value in K-fold cross validation) in search of conclusive evidence. A K value of 10 was used due to the resource- and time-constraints of this thesis.

Next, these findings cannot be immediately extrapolated to ascertain a statement regarding convolutional neural networks and their nodal distributions given an image analysis task. Further research could involve an empirical study to determine the normality of such distributions.

Similarly, further research could involve empirical studies regarding other predictive tasks such as regression rather than classification.

This study evaluated the continuous distributions over neural network nodes rather than parameters. A nodal output is a nonlinear transformation of the sum of its inputs, each of which is a linear transformation on the corresponding parameter, simply scaled by the output of the previous node. It is reasonable to assume that different parameters may have different ideal continuous distributions, so ideal nodal distributions are unlikely to have the same shape as ideal parameter distributions. Thus, further research could involve similar empirical evaluations on the shapes of parameter distributions.

This study did not apply variational dropout techniques to find an ideal distribution shape as it was not deemed necessary to do so in order to demonstrate the validity of non-Gaussian distributions in continuous dropout. Further research could apply variational dropout, or other Bayesian techniques, to learn distribution parameters if reasonably achievable. This would also allow for different nodes or parameters to have their own associated distributions without exploding the dimensions of a hyperparameter grid search, which is likely to result in further improved performance.

Lastly, distributions of entirely different classes could be empirically tested. As the posterior distributions over neural network parameters are intractable, there is no guarantee that unimodal symmetric distributions are necessarily ideal. Perhaps multimodal or even asymmetric distributions may result in improved performance.

VI. Conclusion

Recent work has shown that continuous, Gaussian dropout can perform as well or better than Bernoulli dropout, and that continuous dropout in general can be interpreted as treating neural network parameters as probability distributions rather than point estimates. Other related work has shown that a continuous probability distribution can be fine-tuned through its parameters to minimize the Kullback-Leibler divergence between it and the posterior distribution over the corresponding parameter. However, all empirical studies associated with this recent work has focused on Gaussian distributions.

While it may be reasonable, or at least intuitive, to assume that the true continuous distribution over an arbitrary neural network node or parameter is roughly symmetric, there is no reason to assume that they are necessarily Gaussian. As these distributions do not arise from observation or frequency, but rather are generated in order to maximize the model's performance while minimizing its overfitting, they are entirely intractable and can generally be considered a construct rather than a naturally occurring event. As such, it is unlikely that a Gaussian distribution will necessarily achieve the most optimal results when applying a continuous probability distribution to neural network nodes or parameters.

In this study, various instances of a generalized normal distribution (specifically an exponential power distribution), capable of representing continuous distributions between a zero-centered normal distribution and a zero-centered uniform distribution, were empirically evaluated as nodal probability distributions in their abilities to yield maximal test accuracy within a multilayer perceptron trained against the MNIST and MNIST Fashion datasets. It was shown that the shape parameter of the generalized normal distribution has a statistically significant effect on the performance of the multilayer perceptron when trained against MNIST. The data also seemed to support that optimal performance does not necessarily coincide with convergence toward a Gaussian distribution, nor with convergence toward a uniform distribution, but rather interpolations between the two. However, the data was not sufficiently significant to make any conclusions in this regard. Although variational dropout and other Bayesian techniques were not applied, this is sufficient in demonstrating the validity of constructing non-Gaussian continuous distributions to represent neural network nodes.

References

[1] M. Minsky, "Steps toward artificial intelligence," *Proceedings of the IRE*, vol. 49, pp. 8–30, Jan. 1961.

[2] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals and Systems*, vol. 2, pp. 303–314, Dec. 1989.

[3] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, vol. 15, pp. 315–323, Apr. 2011.

[4] V. Vapnik and A. Chervonenkis, "On the uniform convergence of relative frequencies of events to their probabilities," *Theory of Probability and its Applications*, vol. 16, No. 2, pp. 264–280, 1971.

[5] R. Caruana, S. Lawrence, and L. Giles, "Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping," *Proceedings of the 13th International Conference on Neural Information Processing Systems*, pp. 303–314, Jan. 2000.

[6] G. Hinton *et al.*, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv e-prints*, p. arXiv:1207.0580, Jul. 2012.

[7] N. Srivastava *et al.*, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, Jan. 2014.

[8] D. Kingma, T. Salimans, and M. Welling, "Variational dropout and the local reparameterization trick," *Advances in Neural Information Processing Systems*, vol. 28, 2015.

[9] J. Angus, "The probability integral transform and related results," *Society for Industrial and Applied Mathematics Review*, vol. 36, No. 4, pp. 652–654, Dec. 1994.