Robust Non-linear Normalization of Heterogeneous Feature Distributions with Adaptive Tanh-Estimators

Felip Guimerà Cuevas
BMW Group Munich, Germany
felip.guimera-cuevas@bmw.de

Helmut Schmid
Center for Information and Language Processing
LMU Munich, Germany
schmid@cis.lmu.de

Abstract

Feature normalization is a crucial step in machine learning that scales numerical values to improve model effectiveness. Noisy or impure datasets can pose a challenge for traditional normalization methods as they may contain outliers that violate statistical assumptions, leading to reduced model performance and increased unpredictability. Non-linear Tanh-Estimators (TE) have been found to provide robust feature normalization, but their fixed scaling factor may not be appropriate for all distributions of feature values. This work presents a refinement to the TE that employs the Wasserstein distance to adaptively estimate the optimal scaling factor for each feature individually against a specified target distribution. The results demonstrate that this adaptive approach can outperform the current TE method in the literature in terms of convergence speed by enabling better initial training starts, thus reducing or eliminating the need to re-adjust model weights during early training phases due to inadequately scaled features. Empirical evaluation was done on synthetic data, standard toy computer vision datasets, and a real-world numeric tabular dataset.

1 INTRODUCTION

Knowledge representation (KR) is the process of arranging and organizing information such that a computer can interpret and manipulate it. The goal of KR is to express information, concepts, and associations in a precise manner that a computer can understand. There are several ways to represent data, with feature vectors being one of the most typical for many applications and data types. A feature vector is a numerical representation of a data instance in which each feature relates to a different attribute or characteristic of the instance. Feature scaling (FS), also known as data normalization, is a data pre-processing technique that is used to adjust the range and scope of variables (i.e. features). Effective pre-processing of numeric data and KRs is crucial and fundamental for good model performance, as the ranges of feature values can vary widely, but comparable feature value domains are usually essential for optimal training. When it comes to training time, convergence, and model performance, adequate FS can be the difference between generating a strong or weak model, or even between complete failure and success. Furthermore, especially for neural networks, gradient descent converges much quicker with FS than without it, yet not all FS methods are equally effective.

Different types of data (e.g. numeric tabular data, images, sound waves, time series, text, etc.) typically support different normalization techniques. Non-linear trend removal, for example, is not applicable to textual data. Methods on tabular data, however, are commonly widely applicable since they operate on numerical values; and so, prevalent in many data domains. Re-scaling through min-max normalization, mean/median normalization, standardization (alias Z-score normalization), and scaling to unit length are some of the most popular scaling methods for tabular features. Other non-linear transformations include translations to uniform distributions (e.g. with quantile transformations) or Gaussian distributions (e.g. with power transformations), and polynomial feature generations. There are many more methods, such as data discretization, clipping, log-scaling, etc.

FS is a sub-field of data pre-processing (PP), separate from different related tasks like data cleaning, transfor-

Proceedings of the 27$^{\text{th}}$ International Conference on Artificial Intelligence and Statistics (AISTATS) 2024, Valencia, Spain. PMLR: Volume 238.

mation, integration, noise detection, or missing value imputation [García et al., 2016]. PP and FS are well explored, and many reviews exist [García et al., 2015, Subasi, 2020, Jayalakshmi and Santhakumaran, 2011, Li et al., 2021].

The domain range and distribution of input feature values are crucial in machine learning. Outliers are data points that deviate considerably from the rest of the observations in a particular data collection [Miller, 1993, Chiang et al., 2003, Rousseeuw and Hubert, 2011], e.g. extreme statistical points at the tails of feature distributions or points that come from a completely different distribution (out-of-distribution). Outliers can have a negative impact and mislead the training process, resulting in longer training times, poorer generalization, inferior performance, and the violation of statistical assumptions. There are different types of outliers [Foorthuis, 2018]; classified as univariate or multivariate, i.e. independent or dependent of other features, and further divided into three categories: point anomalies, conditional outliers, and collective outliers.

Tanh-estimator (TE) normalization was initially proposed as a method that suppresses univariate outliers for point anomalies by applying a tanh function in conjunction with a fixed spread value on feature-wise pre-scaled data (Eq.1) [Hampel et al., 2011]; squishing and bounding extreme values inside a desired range. The pre-scaling employs the *Hampel* function (Eq.2) [Hampel, 1974, Shevlyakov et al., 2008] for robust estimation that is not unduly influenced by anomalies, by lowering the importance of points towards distribution tails. A trade-off between robustness and optimality can be imposed by changing the function's hyper-parameters, e.g. to produce stronger or weaker effects on feature distribution tails. The tanh spread-factor directly influences the standard deviation of the pre-scaled feature distributions; stretching or contracting them respectively. The authors employed a constant, fixed spread value for all features, introducing a new parameter that requires manual tweaking. However, incorrect or unfortunate spread values may excessively squish or stretch the value distribution too much, which can be harmful to training. Hence, the idea of having a fixed spread value for all features with possibly rather different distributions is questionable. Also, given large feature input dimensions, manually tuning and estimating the spread value of features can be very time-consuming, inefficient, and impracticable.

This paper presents the Wasserstein tanh-estimator (WTE), an adaptive form of tanh-normalization that automatically estimates the ideal spread value that best minimizes the *Wasserstein distance* (WD) [Panaretos and Zemel, 2019] between a given feature distribution and some target distribution (e.g. a standard Gaussian), calculated for each feature independently. WTE builds and improves upon follow-up work on the tanh-normalization [Latha and Thangasamy, 2011], which applies the tanh directly on the standardized features without applying the *Hampel* estimator for efficiency reasons. WTE can clearly enhance the convergence speed compared to relying on a single fixed default spread value (e.g. of 0.01), particularly in the initial phases of training. Our method's essential qualities are as follows:

- It minimizes the distribution disparity between normalized features and a target distribution to prevent poor or sub-optimal feature distributions.
- It requires no hyper-parameter tuning and finds the ideal spread value via simple scalar minimization methods (e.g. *Brent* [Brent, 1971]).
- The pre-processing can (optionally) be further incorporated into the model's training procedure and optimized accordingly.

For example, it is often advantageous to have feature output distributions that closely resemble a standard Gaussian. WTE can efficiently compute and determine the optimal spread values for all features given a target distribution, enhancing training effectiveness and minimizing the need for weight readjustments caused by otherwise sub-optimal FS.

2 RELATED WORK

Tanh-normalization for FS has been shown to be effective on multiple tasks [Bhanja and Das, 2018, Jain et al., 2005, Ribaric and Fratric, 2005]. It is robust to noise and univariate outliers, but more intricate to compute than simpler methods (e.g. Z-scores). TE was originally defined as [Hampel et al., 2011]:

$$\phi(x) := \frac{1}{2}\left[\tanh\left(0.01\frac{(x-\mu_\Psi)}{\sigma_\Psi}\right)+1\right] \qquad (1)$$

μ_Ψ, σ_Ψ are the mean and standard deviation of the *Hampel* estimator (HE) scores [Hampel, 1974, Shevlyakov et al., 2008]. The HE is a three-part re-descending M-estimator [Hampel, 1973]; it can entirely reject gross outliers while not totally dismissing moderate outliers using the re-descending function Ψ:

$$\Psi(x) = \begin{cases} x, & 0 \le |x| \le a \\ a\ \mathrm{sign}(x), & a \le |x| \le b \\ \frac{a(c-|x|)}{c-b}\ \mathrm{sign}(x), & b \le |x| \le c \\ 0, & c \le |x| \end{cases} \qquad (2)$$

The parameters $a, b, c \in \mathbb{R}$ have an effect on points at the distribution tails and determine the robustness of the estimator.[1] Note that Ψ itself is not the normalization. If the influence of a high number of distribution tail-points is minimized based on the parameters, the estimate becomes more robust to outliers but less efficient (optimal). Here, "efficiency" refers to utilizing data for precise estimates. A less efficient FS may distort the data more but can be more resilient to noise and outliers. In contrast, if the estimate can be influenced by a large number of tail-points, it becomes less robust but more efficient (linear dependencies are better preserved). The tanh-distribution in the transformed domain has a mean of 0.5 and a standard deviation of about 0.01. The spread of the normalized scores is determined by the constant (i.e. 0.01) in the tanh-normalization equation [Jain et al., 2005]. By definition, the range of the tanh-estimator is $[0, 1]$, but it can easily be constraint linearly within the range of $[-1, 1]$ using the normalization of $\phi'(x) := \tanh\left(0.01\frac{(x-\mu_\Psi)}{\sigma_\Psi}\right)$.

Choosing the right *Hampel* parameters is crucial but challenging. Thus, a modified tanh-normalization was proposed [Latha and Thangasamy, 2011] that used the raw features' mean and standard deviation instead of the *Hampel* scores; considerably simplifying and rewriting the normalization of each feature $x \in X$ to $\phi''_X(x) := \tanh(0.01 \cdot \overline{x})$; where $\overline{x}$ is the respective standardized value. This avoids the complexity of the *Hampel* function, resulting in a much more straightforward formula and easier computation. They did, however, still employ a default constant value of 0.01 (which we refer to as the *spread value* in this paper); and can be seen as a configurable hyper-parameter. Thus, although the normalization is now both robust and fast to compute, it still requires correct spread parameter estimates or manual tweaking [Jain et al., 2005]; as also noticed in other work [Atrey et al., 2010], where the constant spread value in the tanh normalization for a fingerprint modality was increased to 0.1. Further issues regarding the spread value were also reported in other use-cases; e.g. where face match scores generated using a multi-layer perceptron classifier and a non-linear tanh function presented problems [Nandakumar et al., 2007]: the outputs peaked at -1 and 1, resulting in poor performance. This shows and underscores the importance of properly pre-scaling features before the tanh function.

The tanh function was also utilized in other different FS approaches. For example, Linear-Tanh-Linear (LTL) [Singh and Gupta, 2007] was presented for biometric systems and is based on the tanh-estimator over a set of scores O_k^G and a set of imposter scores O_k^I for a characteristic k. It maps the non-overlapping region of imposter scores to a constant value of zero and the non-overlapping region of scores to a constant value of one. The tanh-estimator is then used to map the values of the overlapping region between O_k^G and O_k^I via a non-linear tanh function. The authors concluded that LTL is both, efficient and robust; but again, used a fixed spread value of 0.01. Additionally, the tanh-function's resilience to anomalies has also found application in trimmed estimators [Leonowicz et al., 2005] for robust averaging when the number of trials is small and the data is highly non-stationary or contains outliers, and also used to compress over-large values on images to decrease their effect on later stages of processing [Tan and Triggs, 2010]. Lastly, a similar robust normalization method, yet without the use of tanh, is the double sigmoid normalization [Cappelli et al., 2000], which converts scores within a region linearly and scores outside the region non-linearly. However, it also necessitated cautious tweaking of hyper-parameters.

Returning to the implications of selecting a spread value α, there is no justification for why any given fixed α should be optimal for all feature distributions. In essence, it is far more plausible that each feature distribution will have its own (ideal) spread value, and that a spread of e.g. 0.01 may even be harmful to training in many cases. Previous work on tanh-normalization has not sufficiently addressed this concern, and given the impact of FS, a robust and adaptable feature normalizing approach is highly desirable. Therefore, this work addresses the challenge by calculating the feature-wise *ideal* spread value $\dot{\alpha}$ and building on previous work which computes the mean and variance of the features without the *Hampel* estimator and directly performs standard feature-based standardization before the tanh function [Latha and Thangasamy, 2011]. That is, instead of relying on a predetermined global fixed default spread, this paper proposes a better solution that directly identifies the best $\dot{\alpha}$ for each feature in $\tanh(\dot{\alpha}\overline{x})$. In particular, we discuss and elaborate on the key role of α in improving training convergence.

3 BACKGROUND

By replacing the default value of 0.01 with α, an adjustable parameter is introduced. The correct choice of α is critical since it regulates the spreading of the output values prior to the non-linear tanh transformation and, thus, affects the distributions of the normalized feature values. We might want to keep these distributions close to e.g. a Gaussian. To measure how much a given distribution deviates from a desired distribution,

[1] Ψ functions are non-decreasing near the origin, but decreasing (towards zero) far from the origin. Andrew's Sine Function [Hinkley, 1973] and Tukey's Biweight Function [Beaton and Tukey, 1974] are alternatives to Hampel's Ψ function. Redescending M-estimators to handle outliers are also used in regression contexts [Khan et al., 2021].

we can use the WD. Formally, the p-th WD between two distributions u, v on $\mathbb{R}$ is defined as:

$$W_p(u,v) := \inf_{\pi \in \Pi(u,v)} \left(\int_{\mathbb{R}\times\mathbb{R}} d(x,y)^p \, \mathrm{d}\pi(x,y) \right)^{1/p} \quad (3)$$

where Π is the set of joint distributions π for pairs $(x \sim u, y \sim v)$ with marginals u, v. For $p := 1$ and $d(x,y) := |x-y|$, it follows [Ramdas et al., 2017]:

$$\begin{aligned} W_1(u,v) &= \inf_{\pi \in \Pi(u,u)} \int_{\mathbb{R}\times\mathbb{R}} |x-y| \, \mathrm{d}\pi(x,y) \\ &= \int_{-\infty}^{\infty} |U(t) - V(t)| \, \mathrm{d}t \end{aligned} \quad (4)$$

where U, V are the cumulative distribution functions (CDF) of u, v respectively. Informally, the WD metric is considered the smallest "cost" of converting one probability distribution to another.

4 METHODS

Without loss of generality, we restrict feature value normalization within the interval $[-1, 1]$; aligned with the domain of tanh. Transformations to other domains (e.g. $[0, 1]$) can be easily obtained by simple linear transformations. This work adopts and builds upon the formula for the modified tanh-normalization (MTN) [Latha and Thangasamy, 2011] which applies tanh on standardized features, and omits the *Hampel* function. Spread values are considered *ideal* if they minimize the target WD loss (Eq.7).

4.1 Feature Mapping

Under the premise that the optimal distribution of feature values follows a target distribution with desired properties (e.g. zero mean, unit Gaussian variance [LeCun et al., 2012]) the MTN formula is *not* ideal due to a global constant spread value for all features. Let μ_X, σ_X be, respectively, the mean and standard deviation of the values $x \in \mathbb{R}$ of a feature X with an arbitrary but fixed (discrete) distribution. The fixed spread value can be replaced by a tunable α:

$$\phi_X^\alpha(x) := \tanh\left(\frac{\alpha(x - \mu_X)}{\sigma_X} \right) \quad (5)$$

The goal is to find the best α for each feature such that the CDF $F_{\phi_X^\alpha}$ of the discrete output distribution ϕ_X^α approximates that of a target distribution. Exemplary, this work will use a standard Gaussian as the target distribution. Let F_Z be the CDF of the standard Gaussian $f_Z(x) := \frac{e^{-x^2/2}}{\sqrt{2\pi}}$. The objective loss $\mathcal{L}_W$ is defined per feature via the first WD and Eq.4 as:

$$\mathcal{L}_W(\alpha) := \int_{-\infty}^{\infty} |F_Z(t) - F_{\phi_X^\alpha}(t)| \, \mathrm{d}t \quad (6)$$

By minimizing the WD in Eq. 6 wrt. the spread value α, we make the normalized feature distribution most similar to the target distribution (e.g. the Gaussian). However, the normalized tanh feature values are constrained to the range $[-1, 1]$, whereas the Gaussian is defined for all real values. One can here modify the probability density function (PDF) of f_z for values < -1 and > 1 to be zero and allow a better comparison without considering tail-distributions; or use a truncated[2] Gaussian f_Z^q at a quantile $0 < q < 1$, scaled to a domain of $[-1, 1]$. Low q values have greater min/max values, which squish the mass density towards the center, whereas high values distribute the mass more towards the tails. The parameter q, thus, serves to determine the probability mass threshold of the truncation. By varying the standard deviation of the truncated Gaussian, one can choose distributions where the probability mass is either concentrated in the middle or distributed more evenly. To assure equivalent domain ranges (i.e. $[-1, 1]$), let the PDF $\hat{f}_Z^q(x) := \frac{2(f_Z^q(x) - min(f_Z^q))}{max(f_Z^q)} - 1$. $\hat{F}_Z^q(t)$ is the respective CDF function. The new truncated loss is:

$$\mathcal{L}_W^q(\alpha) := \int_{-1}^{1} |\hat{F}_Z^q(t) - F_{\phi_X^\alpha}(t)| \, \mathrm{d}t \quad (7)$$

Direct minimization of $\mathcal{L}_W^q$ over α is not convex, as can be seen in Figure 5, where convexity is broken for several different feature distributions around $\alpha \approx 0.6$. However, when a normal Gaussian is used as the target distribution (and also for several other distributions), we observe that the loss function $\mathcal{L}_W^q(\alpha)$ is quasi-convex (Figure 5), which implies a single (global) minimum. Thus, to calculate the optimal spread per feature given q, the efficient numerical optimization method *Brent*[3] [Brent, 1971] can be leveraged. Alternatively, other gradient-free algorithms [Brent, 2013] may also be used. Because this pre-processing step will only be executed once, its performance is not critical. For a specific feature, its optimal spread value $\dot{\alpha}$ is the spread α that minimizes the WD loss $\mathcal{L}_W^q(\alpha)$ for a given the target distribution, given by:

$$\dot{\alpha} := \operatorname*{argmin}_{\alpha} \mathcal{L}_W^q(\alpha) \quad (8)$$

[2] Let *ppf* be the percent point function (alias quantile function) of f_Z such that $F_Z\big(ppf(q)\big) = q$. For $x \in [ppf(q), ppf(1-q)]$; $f_Z^q(x)$ is the truncated Gaussian.

[3] *Brent* uses a fast-converging secant technique or inverse quadratic interpolation, but if necessary, adopts a more robust bisection approach.

4.2 Ideal Spread Value

The *ideal* spread values (Eq.5) are incorporated into the MTN formula [Latha and Thangasamy, 2011], wherein the static spread value of e.g. 0.01 is replaced by the corresponding ideal $\dot{\alpha}$ of each feature. Consequently, the ideal non-linear tanh transformation function for each individual feature becomes:

$$h_{\dot{\alpha}}(x) := \tanh\left(\frac{\dot{\alpha}(x-\mu_X)}{\sigma_X}\right) \tag{9}$$

4.3 Trainable Spread Value

In many machine learning domains, one can here additionally learn a training-objective-specific spread value γ (and fine-tune $\dot{\alpha}$) as part of the training process, e.g. via back-propagation optimization. Let $\tau > 0$ be a trainable parameter. Then, a corresponding *trainable* tanh-estimator h_γ can be defined by simply setting $\gamma := \tau\dot{\alpha}$, and so, $h_\gamma(x) := \tanh\left(\frac{\tau\dot{\alpha}(x-\mu_X)}{\sigma_X}\right)$. By adjusting τ, we can learn the ideal spread of the feature distribution during training end-to-end. Because the backward optimization here is solely dependent on τ, $\dot{\alpha}$ is still only computed once before the training and can be reused for different training runs. The model can now "learn" which normalization (i.e. spread value) is the best for optimizing a specific model loss and training objective; also in conjunction with other features. Recall that otherwise $\dot{\alpha}$ is determined independently of other features and specific training goals.

5 EXPERIMENTAL METHODS

We used several common statistical distributions for analysis, including an Alpha-Gamma distribution with parameters $\alpha = \gamma = 4.0$, a generalized Pareto distribution with parameter $\xi = 1$, a symmetric bi-modal distribution with parameters location=-3, scale=1, size=10, etc. We further created an artificial synthetic classification set of size 10,000 with ten classes. Each class in the synthetic set consisted of two Gaussian distributed clusters spanning along the vertices of a hyper-cube. To introduce covariance, features were independently drawn and then randomly (linearly) combined within each cluster. Outliers are restricted to mixtures of Gaussian distributions, which may be a limitation here. Additionally, 5% of the labels were randomly assigned, introducing multivariate outliers. A small feed-forward neural network with four hidden layers, each containing 64 neurons, was used to classify the synthetic data. The Gaussian Error Linear Unit was chosen as the activation function. Synthetic data sets are useful for benchmarking because they offer control of data properties and allow quick and deliberate performance testing. WTE was also evaluated on four commonly used toy computer vision datasets (Fashion-MNIST, EMNIST, CIFAR10, and CIFAR100) using ResNet9. Pixel values were normalized after applying a gray-scale conversion; images were re-scaled to a size of 32x32. We used classification performance over time as the evaluation metric. As a real-world dataset, CDC Diabetes Health Indicators [Teboul, 2023] was used. For a fair and accurate evaluation, each training iteration involved initializing the model weights identically for all different FS methods. That is, model weight initialization was identical for all methods within a training run but different between different runs. The results were averaged across five runs. Cosine annealing was used as the learning rate scheduler, starting at 0.001.

6 RESULTS

6.1 Classification With Neural Networks

We evaluated the effectiveness of neural networks in classifying synthetic data (Figure 1), computer vision toy data (Figure 2), and real-world data (Figure 3); for different spread values. The features of synthetic and real-world data were normalized individually. Pixel-normalization for vision sets was on the gray-scale channel. The goal was to examine the impact of FS on the training for different tasks. Specifically, we show that inadequate spread initialization in TEs can harm training convergence; which highlights the importance of choosing a proper normalization spread value.

Training converged much faster with feature-wise ideal spread values $\dot{\alpha}$ than with a fixed global default spread value for all features of $\alpha = 0.01$, as expected. On synthetic tabular data (points drawn from linear Gaussian combinations), an $\alpha = 1$ exhibited a similar training convergence as $\dot{\alpha}$, but the value range was too large. Notice, how $\alpha = 1$ is equivalent to directly applying tanh to the standardization. Conversely, using a global default spread of $\alpha = 0.01$ achieved the worst performance on the synthetic dataset, and considerably constrained the effective feature range. Overall, the probability mass of the KDE of the normalization for $\dot{\alpha}$ produced the best distribution, with all features having similar peak densities and being Gaussian-like inside $[-1, 1]$. On the computer vision sets, $\dot{\alpha}$ demonstrated substantially less performance variance between independent runs. Here, although the ideal spread values $\dot{\alpha}$ clearly improved training convergence speed, the final scores towards the end of the training were about the same for all α values. This suggests that a good choice of α strongly contributes to a better training start and reduces or eliminates the need to adjust training weights to poorly chosen feature distribution. Furthermore, recall that, while features are

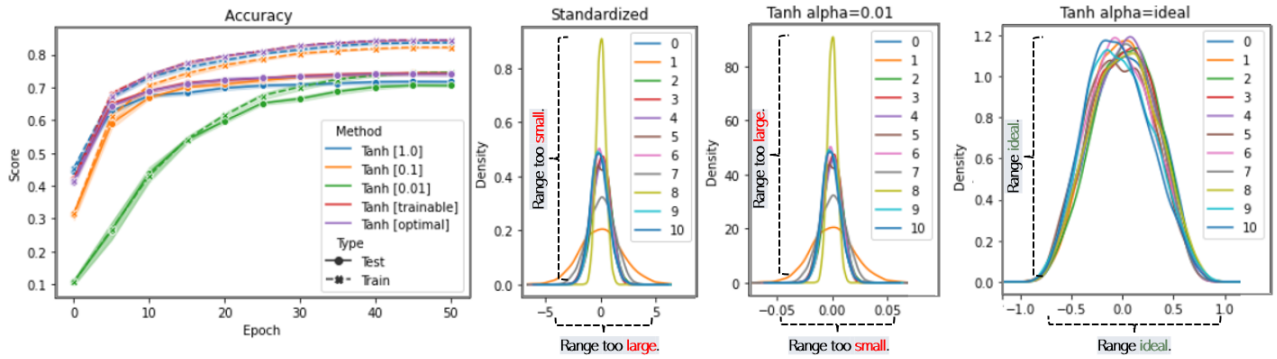

Figure 1: Comparing normalized KDEs and model performance on synthetic data. Training accuracy is depicted in the left figure; the others show the KDEs of the normalized features. A default spread of $\alpha = 0.01$ showed the worst convergence. $\dot{\alpha}$ resulted in the best probability mass distribution.

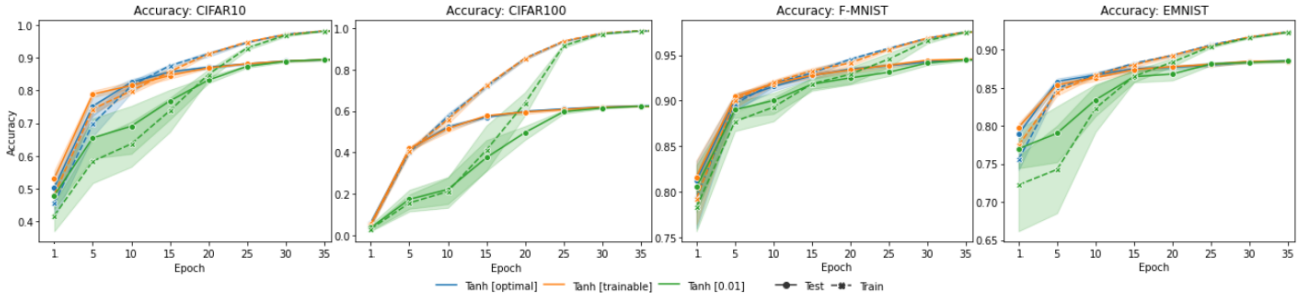

Figure 2: Comparing the training performance of ideal, trainable, and fixed spread values on four different toy computer vision datasets after tanh-normalizing the gray-scale pixel value distribution.

initially normalized independently, back-propagation (optionally) allows us to alter each spread value in relation to the other features. Yet, the performance of $h_{\dot{\alpha}}$ and h_{γ} for a trainable $\gamma := \tau\dot{\alpha}$ was nearly equal and did not show any noticeable performance gains. The motivation here was that, since $\dot{\alpha}$ is calculated based on a particular target distribution, the optimal FS for a given task might require a different target distribution, e.g. one that is not Gaussian. By allowing the parameter τ to adjust (i.e. be learned), the spread value can further be adapted towards the "optimal" spread for a given task. Lastly, we analyzed the classification performance of different spread values on the larger real-world CDC Diabetes Health Indicators dataset (Figure 3). Again, the adaptive normalization outperformed the use of a fixed global spread value in terms of convergence speed. But as before, fine-tuning $\tau\dot{\alpha}$ did not substantially improve performance.

Overall, our primary finding was that slow convergence rates in tanh normalizers were primarily due to poorly dispersed density masses in the scaled feature distributions. Inadequate spread values resulted in bad density dispersion and forced initial model weights to adjust to the input domain, thereby slowing down convergence speed. Thus, a fixed global spread value is sub-optimal; but ideal spreads are easily determined.

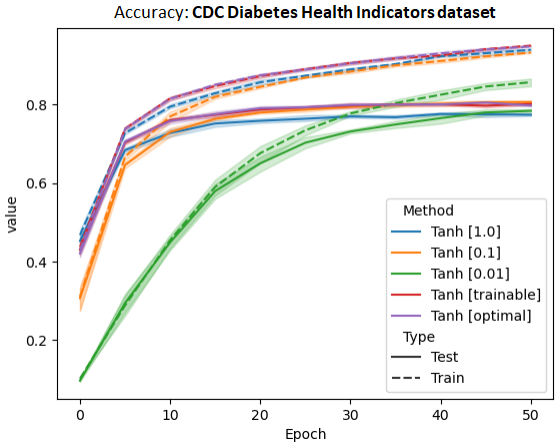

Figure 3: Classification scores on the CDC Diabetes Health Indicators dataset for different spread values.

6.2 Analytical Optimization

Although the probability mass of a Gaussian is distributed symmetrically following a tanh transformation, a default spread of $\alpha = 1$ is not suitable since the output probability density places too much weight

on the tails. The transformation might be even worse for other standardized (but non-Gaussian) inputs. Instead of being scattered at the extremes, the bulk of the output probability mass should be spread around the center. That is, having the majority of the mass in the middle and just a small amount (e.g. outliers) squeezed on the tails. The spread parameter of the tanh mapping can be changed to tweak this trade-off. For example, if α is excessively large, the absolute value of practically all features will be extremely close to one. Similarly, if α is too low, the majority of values will be close to zero. Both situations are undesirable because they contradict the purpose of FS. As a result, the target distribution helps to disperse the probability mass appropriately. Figure 4 below shows the kernel density estimate [4] (KDE) of a truncated min-max normalized target Gaussian with $q = 0.001$ following the tanh-transformation, i.e. the KDE of h_α, for various spread values α. In particular, for data X, the kernel estimate analyzed is $\text{KDE}(h_\alpha(X))$. Notice that $\text{KDE}(X) \neq h_\alpha(X)$. The spread value has a strong influence on the KDE of the tanh normalization, with very low spread values strongly squishing the probability mass towards the center.

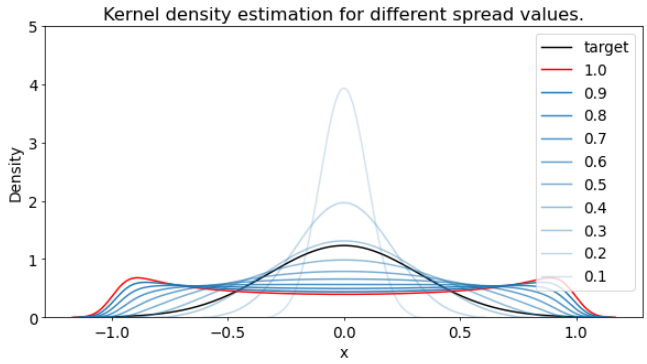

Figure 4: A truncated standard Gaussian with q=0.001 (black) is compared to the KDE of its tanh-normalization output for various α spread values (blue). The result for $\alpha = 1$ is shown in red.

Yet, often, inputs are not normally distributed. Standardization assures a mean and standard deviation of zero and one, respectively, but does not change the underlying (histogram) distribution, i.e. the distribution is relocated and stretched/shrunk, but not "distorted". By minimizing the loss, we merely reduce the mismatch between the output's KDE and a desired target distribution. In other words, we find the spread value α which results in the lowest possible WD loss after applying the tanh normalization. It is worth noting that even if we apply additional *linear* transformations afterward on the output, such as constraining the range to [0,1] instead of [-1,1], the optimal spread value $\dot{\alpha}$ remains the same, but the WD loss may vary. Figure 5 below depicts the WD losses of different distributions for different spread values. Empirically, it appears that, for a standard Gaussian as the target, minimizing over the spread value α may be quasi-convex and feature a unique (global) minimum. The loss itself is, however, not convex. The loss function is also not symmetric, despite point symmetry of tanh; remember, it is non-linear. Hence, adding or subtracting a spread value $\epsilon \pm \dot{\alpha}$ for a distance $\dot{\alpha} > \epsilon > 0$ has a different impact. E.g., the WD may increase "left" from $\dot{\alpha}$ more strongly than "right" of it.

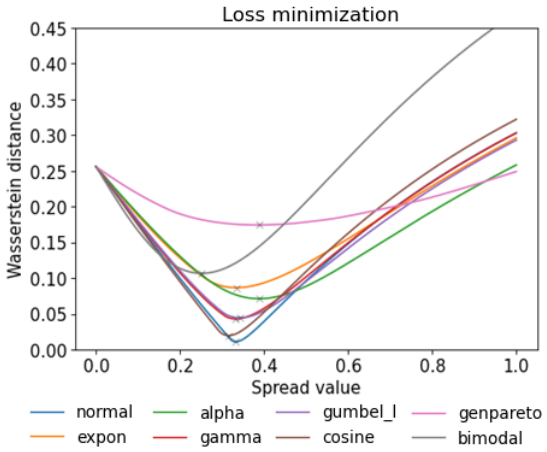

Figure 5: Wasserstein loss against different spread values for various standardized distributions.

A standard (non-truncated) Gaussian has $\approx 68.27\%$ of its probability density within a range of $[-1, 1]$, $\approx 27.18\%$ for $[-2, -1) \cup (1, 2]$; $\approx 4.28\%$ for $[-3, -2) \cup (2, 3]$; and $\approx 0.27\%$ elsewhere. For tanh-normalization, the percentages are subject to the spread value. Given a standard normal distributed input, for $\alpha = 1$ more than two-thirds of the transformed features will have *absolute* values less than 0.762, and more than a quarter will have *absolute* values in $[0.762, 0.964]$. Similarly, the default spread value from the literature of 0.01 [Jain et al., 2005, Latha and Thangasamy, 2011], on the other hand, would here squish more than 99.8%. This raises the question of why such a convention has persisted as this can be considered a terrible scale for feature normalization and may explain why practitioners would resort to different alternative (manually adjusted) spread values. In fact, $\alpha = 0.01$ was initially chosen for a transformation using Hampel, rather than for the standardized features in MTN, but manual tweaking and correction were even for Hampel still involved and often necessary [Jain et al., 2005].

Table 1 shows the probability density coverage for var-

[4] An estimation that uses a continuous probability density curve to describe the data distribution.

ious spread values. Entries represent the supremum of the *absolute* output value for the tanh normalization for a certain percentage of the probability density. The target distribution is a standard Gaussian distribution.

Table 1: Approximate probability density coverage.

α	68.2%	95.4%	99.8%	100%
0.01	0.0010	0.0020	0.0030	1
0.10	0.0997	0.1974	0.2913	1
0.25	0.2449	0.4621	0.6351	1
0.50	0.4621	0.7616	0.9051	1
0.75	0.6351	0.9051	0.9780	1
1.00	0.7616	0.9640	0.9951	1

It emphasizes the issue of selecting spreads that are too small, as the maximum absolute values remain extremely low. Too large spread values, however, were not as detrimental, but still far from ideal.

Close to zero, the tanh function is almost linear, with non-linearity more apparent on larger (absolute) values. A very small spread value, yielding a very small effective range, introduces little non-linearity. Consequently, apart from excessively compressing feature values, this minimal non-linearity fails to diminish the impact of outliers or enhance robustness against noise. Weights will have to re-adapt throughout training, but linearity within individual features is largely "preserved". Large spread values, on the other hand, introduce more non-linearity and greater affect the normalization's distribution. If non-linearity is imposed too strongly, potentially assumed linear relationships (e.g. on the weighted sum) may become distorted. Overall, already values ≤ 0.1 exhibit a considerable WD divergence, and so the tanh estimator's default spread of 0.01 is a very unfortunate choice here. Table 2 contains an overview of spread values and their numerical loss for different ground distributions. The ideal spread value for a Gaussian target distribution was often similar across most distributions; yet not identical.

Table 2: Ideal spread values.

DISTRIBUTION	$\approx \dot{\alpha}$	$\mathcal{L}_W^{q=0.001}$
Normal	0.3328	0.011
Exponential	0.3336	0.087
Alpha	0.3885	0.071
Gamma	0.3326	0.043
Gumbel_L	0.3422	0.044
Cosine	0.3153	0.020
GenPareto	0.3873	0.174
Bimodal	0.2516	0.107

7 DISCUSSION

Pre-processing of feature values is important to ensure optimal model performance. For numeric feature vector representations, it is crucial to ensure that the ranges of feature values are comparable so that no single feature dominates the others; which e.g. could cause bias. FS is often used here to remap the individual features into comparable (unit) ranges. It can, however, be sensitive to outliers in the data and even outliers in the individual feature dimensions. *Tanh*-estimators offer robust feature normalization. They utilize a fixed scaling factor for all features, which raises the question of whether this is suitable given the vast variation of possible feature distributions.

In a series of tests, poorly spread density masses of normalized feature value distributions were to blame for slow training convergence. When a fixed global spread value is used to normalize all features, this is expected to happen, particularly e.g. if feature distributions vary greatly. This work proposed an adaptive form of the tanh-normalization that calculates the ideal spread based on minimizing the Wasserstein distance of feature distributions against a target distribution to control the probability density of the normalization. The method particularly enhances the literature's current method, boosting training convergence speed by avoiding unnecessary model weight adaption due to poor FS. This paper repeatedly highlighted the importance of determining ideal *feature-wise* spread values and provided theoretical motivation. Our solution determines the ideal spreads automatically and effectively, avoiding time-consuming manual adjustments. Just as the traditional tanh-normalization, it is robust to outliers, efficient, and very simple to incorporate, but further ensures a normalizing transformation that gives an effective probability mass distribution.

8 CONCLUSION

For machine learning models to be effective, proper feature pre-processing (FP) is fundamental. The choice of the FP method holds great importance, as inadequate FP can have a (strong) negative impact on training convergence and model performance, especially in machine learning. Although Tanh-Normalization provides outlier resilience, it does so by using a fixed global spread value for all features during the FP; which is not optimal. Instead, a better approach is to choose a spread value that aligns the FP output best with a desired, e.g. well-distributed, target distribution. The ideal tanh spread value is calculated automatically. This not only improves model training convergence noticeably but also eliminates the need for time-consuming manual parameter tuning.

References

[Atrey et al., 2010] Atrey, P. K., Hossain, M. A., El Saddik, A., and Kankanhalli, M. S. (2010). Multimodal fusion for multimedia analysis: a survey. *Multimedia systems*, 16(6):345–379.

[Beaton and Tukey, 1974] Beaton, A. E. and Tukey, J. W. (1974). The fitting of power series, meaning polynomials, illustrated on band-spectroscopic data. *Technometrics*, 16(2):147–185.

[Bhanja and Das, 2018] Bhanja, S. and Das, A. (2018). Impact of data normalization on deep neural network for time series forecasting. *arXiv preprint arXiv:1812.05519*.

[Brent, 1971] Brent, R. P. (1971). An algorithm with guaranteed convergence for finding a zero of a function. *The Computer Journal*, 14(4):422–425.

[Brent, 2013] Brent, R. P. (2013). *Algorithms for minimization without derivatives*. Courier Corporation.

[Cappelli et al., 2000] Cappelli, R., Maio, D., and Maltoni, D. (2000). Combining fingerprint classifiers. In *International Workshop on Multiple Classifier Systems*, pages 351–361. Springer.

[Chiang et al., 2003] Chiang, L. H., Pell, R. J., and Seasholtz, M. B. (2003). Exploring process data with the use of robust outlier detection algorithms. *Journal of Process Control*, 13(5):437–449.

[Foorthuis, 2018] Foorthuis, R. (2018). A typology of data anomalies. In *International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, pages 26–38. Springer.

[García et al., 2015] García, S., Luengo, J., and Herrera, F. (2015). *Data preprocessing in data mining*, volume 72. Springer.

[García et al., 2016] García, S., Ramírez-Gallego, S., Luengo, J., Benítez, J. M., and Herrera, F. (2016). Big data preprocessing: methods and prospects. *Big Data Analytics*, 1(1):1–22.

[Hampel, 1973] Hampel, F. R. (1973). Robust estimation: A condensed partial survey. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 27(2):87–104.

[Hampel, 1974] Hampel, F. R. (1974). The influence curve and its role in robust estimation. *Journal of the american statistical association*, 69(346):383–393.

[Hampel et al., 2011] Hampel, F. R., Ronchetti, E. M., Rousseeuw, P. J., and Stahel, W. A. (2011). *Robust statistics: the approach based on influence functions*, volume 196. John Wiley & Sons.

[Hinkley, 1973] Hinkley, D. V. (1973). Robust estimates of location: Survey and advances.

[Jain et al., 2005] Jain, A., Nandakumar, K., and Ross, A. (2005). Score normalization in multimodal biometric systems. *Pattern recognition*, 38(12):2270–2285.

[Jayalakshmi and Santhakumaran, 2011] Jayalakshmi, T. and Santhakumaran, A. (2011). Statistical normalization and back propagation for classification. *International Journal of Computer Theory and Engineering*, 3(1):1793–8201.

[Khan et al., 2021] Khan, D. M., Ali, M., Ahmad, Z., Manzoor, S., and Hussain, S. (2021). A new efficient redescending m-estimator for robust fitting of linear regression models in the presence of outliers. *Mathematical Problems in Engineering*, 2021.

[Latha and Thangasamy, 2011] Latha, L. and Thangasamy, S. (2011). Efficient approach to normalization of multimodal biometric scores. *International Journal of Computer Applications*, 32(10):57–64.

[LeCun et al., 2012] LeCun, Y. A., Bottou, L., Orr, G. B., and Müller, K.-R. (2012). Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer.

[Leonowicz et al., 2005] Leonowicz, Z., Karvanen, J., and Shishkin, S. L. (2005). Trimmed estimators for robust averaging of event-related potentials. *Journal of neuroscience methods*, 142(1):17–26.

[Li et al., 2021] Li, B., Wu, F., Lim, S.-N., Belongie, S., and Weinberger, K. Q. (2021). On feature normalization and data augmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12383–12392.

[Miller, 1993] Miller, J. N. (1993). Tutorial review—outliers in experimental data and their treatment. *Analyst*, 118(5):455–461.

[Nandakumar et al., 2007] Nandakumar, K., Chen, Y., Dass, S. C., and Jain, A. (2007). Likelihood ratio-based biometric score fusion. *IEEE transactions on pattern analysis and machine intelligence*, 30(2):342–347.

[Panaretos and Zemel, 2019] Panaretos, V. M. and Zemel, Y. (2019). Statistical aspects of wasserstein distances. *Annual review of statistics and its application*, 6:405–431.

[Ramdas et al., 2017] Ramdas, A., Trillos, N. G., and Cuturi, M. (2017). On wasserstein two-sample testing and related families of nonparametric tests. *Entropy*, 19(2):47.

[Ribaric and Fratric, 2005] Ribaric, S. and Fratric, I. (2005). A matching-score normalization technique for multimodal biometric systems. In *Proceedings of Third COST 275 Workshop-Biometrics on the Internet*, pages 55–58. University of Hertfordshire.

[Rousseeuw and Hubert, 2011] Rousseeuw, P. J. and Hubert, M. (2011). Robust statistics for outlier detection. *Wiley interdisciplinary reviews: Data mining and knowledge discovery*, 1(1):73–79.

[Shevlyakov et al., 2008] Shevlyakov, G., Morgenthaler, S., and Shurygin, A. (2008). Redescending m-estimators. *Journal of Statistical Planning and Inference*, 138(10):2906–2917.

[Singh and Gupta, 2007] Singh, Y. N. and Gupta, P. (2007). Quantitative evaluation of normalization techniques of matching scores in multimodal biometric systems. In *International Conference on Biometrics*, pages 574–583. Springer.

[Subasi, 2020] Subasi, A. (2020). Chapter 2 - data preprocessing. In Subasi, A., editor, *Practical Machine Learning for Data Analysis Using Python*, pages 27–89. Academic Press.

[Tan and Triggs, 2010] Tan, X. and Triggs, B. (2010). Enhanced local texture feature sets for face recognition under difficult lighting conditions. *IEEE transactions on image processing*, 19(6):1635–1650.

[Teboul, 2023] Teboul, A. (2023). Diabetes health indicators dataset. `https://www.kaggle.com/datasets/alexteboul/diabetes-health-indicators-dataset`.

Checklist

1. For all models and algorithms presented, check if you include:
   (a) A clear description of the mathematical setting, assumptions, algorithm, and/or model. [Yes]
   (b) An analysis of the properties and complexity (time, space, sample size) of any algorithm. [Not Applicable]
   (c) (Optional) Anonymized source code, with specification of all dependencies, including external libraries. [Yes]

2. For any theoretical claim, check if you include:
   (a) Statements of the full set of assumptions of all theoretical results. [Yes]
   (b) Complete proofs of all theoretical results. [Not Applicable]
   (c) Clear explanations of any assumptions. [Not Applicable]

3. For all figures and tables that present empirical results, check if you include:
   (a) The code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL). [Yes]
   (b) All the training details (e.g., data splits, hyperparameters, how they were chosen). [Yes]
   (c) A clear definition of the specific measure or statistics and error bars (e.g., with respect to the random seed after running experiments multiple times). [Yes]
   (d) A description of the computing infrastructure used. (e.g., type of GPUs, internal cluster, or cloud provider). [Not Applicable]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets, check if you include:
   (a) Citations of the creator If your work uses existing assets. [Yes]
   (b) The license information of the assets, if applicable. [Yes]
   (c) New assets either in the supplemental material or as a URL, if applicable. [Yes]
   (d) Information about consent from data providers/curators. [Not Applicable]
   (e) Discussion of sensible content if applicable, e.g., personally identifiable information or offensive content. [Not Applicable]

5. If you used crowdsourcing or conducted research with human subjects, check if you include:
   (a) The full text of instructions given to participants and screenshots. [Not Applicable]
   (b) Descriptions of potential participant risks, with links to Institutional Review Board (IRB) approvals if applicable. [Not Applicable]
   (c) The estimated hourly wage paid to participants and the total amount spent on participant compensation. [Not Applicable]