Generating Heavy-Tailed Synthetic Data with Normalizing Flows

Saba Amiri[1] **Eric Nalisnick**[1] **Adam Belloum**[1] **Sander Klous**[1] **Leon Gommans**[2]

[1]Informatics Institute, University of Amsterdam
[2]Air France - KLM

Abstract

Heavy-tailed data is commonly encountered but difficult to model well. We experimentally compare the ability of three different normalizing flows to model data with varying tail behavior. The flows are parameterized using base densities with differing tail properties. We report results on both simulations and a real-world data synthesis task.

1 INTRODUCTION

There are often challenges that come along with training models on real data. For instance, privacy concerns may require that the original observations be obscured. Or perhaps the original data set is extremely large, storing and transporting it may be prohibitively costly. *Data synthesis* [Rubin, 1993] aims to address these problems by producing proxy data from which nearly identical inferences can be drawn. This proxy data is often generated by a model, which forms a compact representation of the original data and can produce additional data on demand.

We turn our attention to a particularly challenging subproblem in data synthesis: generating heavy-tailed data. Many real-world settings produce heavy-tailed data—such as traffic in communication networks, risk in actuarial analysis, and cumulative damage in survival analysis—but unfortunately, fitting the appropriate models can be difficult. For instance, the Cauchy distribution has undefined moments. Moreover, we would like our models to be tail-adaptive in the sense that they can automatically adjust themselves to have the appropriate tail behavior—anywhere between Cauchy-like and sub-Gaussian tails.

In this work, we investigate the ability of *normalizing flows* (NFs) [Tabak and Turner, 2013] to perform heavy-tailed data synthesis. NFs provide an attractive model class for this exploration since (i) they have demonstrated powerful generative abilities (e.g. on high-resolution images and audio) and (ii) admit some analytical understanding of their tail properties [Jaini et al., 2020]. We experimentally compare NFs with normal, generalized normal, Student's t, and mixture (of normals) base densities, finding that the mixture performs the best, possibly due to its more stable optimization.

2 BACKGROUND

Notation We denote random variables with bold letters and observations with non-bold letters. We denote data by $\boldsymbol{x}$ and its observation by $\mathbf{x}$. The kth component of $\mathbf{x}$ is denoted by x_k.

Normalizing Flows *Normalizing flows* (NFs) [Tabak and Turner, 2013, Rezende and Mohamed, 2015, Papamakarios et al., 2021] are deep generative models built from the principle of reparameterization. A random variable, usually following a simple distribution (e.g. normal), is pushed through a series of bijective, neural-network-based transformations. The resulting transformed random variable follows a much richer distribution (compared to the one pre-transformation) while still having a tractable density function. We denote the N-step transformation $T_\phi = T_{N-1} \circ \ldots \circ T_0$, where ϕ denotes the neural network parameters. Let $\boldsymbol{u} \sim p_u(\boldsymbol{u})$ denote the base density that will undergo the transformation $\boldsymbol{x} = T(\boldsymbol{u})$. We assume $\boldsymbol{x}$ denotes the data, and thus given an observation $\mathbf{x}$, we can evaluate its density using the change of variables formula:

$$p_\phi(\mathbf{x}) = p_u\left(T_\phi^{-1}(\mathbf{x})\right)\left|\det J_{T_\phi^{-1}}(\mathbf{x})\right|$$

where $J_{T_\phi^{-1}}$ is the Jacobian matrix of inverse transformation. Training the model parameters ϕ can be done via maximum likelihood estimation using the same density function. After training, samples can be drawn using the forward transform: $\hat{\mathbf{u}} \sim p_u$, $\hat{\mathbf{x}} = T_\phi(\hat{\mathbf{u}})$.

Accepted for the 5th Workshop on Tractable Probabilistic Modeling at UAI (TPM 2022).

Related Work on Data Synthesis Much of the previous work on using deep generative models for data synthesis has focused on *generative adversarial networks* (GANs) [Goodfellow et al., 2014]. While this work has shown promise, we believe that GANs are not appropriate for heavy-tailed data since there is no known way to understand or control their tail behavior. Moreover, GANs are prone to underestimate the support of the target distribution ("mode collapse"), and this behavior would only be exacerbated by heavy tails. NFs have been previously explored for data synthesis by Kamthe et al. [2021], but they did consider the specific case of heavy-tailed data.

Related Work on Heavy-Tailed Flows Jaini et al. [2020] analyzed the tail behavior of NFs. They show that NFs built from Lipschitz-continuous transforms will always have a target density with the same tail properties as the base distribution. This limitation applies to many popular NF architectures (e.g. RNVP [Dinh et al., 2017], Glow [Kingma and Dhariwal, 2018]), and Jaini et al. [2020] propose making the NF tail-adaptive by estimating parameters in the base density—in particular, the degrees of freedom of a Student's t. Jaini et al. [2020]'s contribution is mainly theoretical, and they did not investigate how best to implement tail-adaptive NFs in practice.

3 HEAVY-TAILED NORMALIZING FLOWS

We now summarize the three types of heavy-tailed NFs we will use in the experiments. In all cases, we use the *real non-volume preserving* (RNVP) architecture: a series of affine coupling layers interleaved with permutation operations (over dimensions). RNVP is a Lipschitz-continuous function, and thereby the resulting density function has the same tail properties as the base distribution. Thus, we consider a range of base densities:

1. **Student's t**: Following Jaini et al. [2020]'s recommendation, we consider the multivariate Student's t-distribution, which has the density function:

$$p_u(u) = \frac{\Gamma((\nu+1)/2)}{\sqrt{\nu\pi}\,\Gamma(\nu/2)} \left(1 + \frac{u^2}{\nu}\right)^{\frac{\nu+1}{2}}$$

   where ν is the degree of freedom parameter. The ν parameter allows the Student's t to interpolate between the Cauchy ($\nu = 1$) and normal ($\nu \to \infty$) distributions, with the former having the heaviest tails.

2. **Generalized Normal**: We consider a variant of the normal distribution that includes a parameter that controls tail behavior. The *generalized normal* distribution has the desnity function:

$$p_u(u) = \frac{\beta}{2\alpha\Gamma(1/\beta)} \exp\left\{-\left(\frac{|u|}{\alpha}\right)^{\beta}\right\}$$

   where $\alpha \in \mathbb{R}^+$ is the scale, and $\beta \in \mathbb{R}^+$ is the shape. The shape β controls the tail behavior. As β approaches 0, the tails become heavy, and as β approaches infinity, the density converges to the continuous uniform on $(\pm\alpha)$. When $\beta = 2$, the normal is recovered, and when $\beta = 1$, the density is that of the Laplace distribution. There are two primary differences between the Student's t and generalized normal. The first is the generalized normal can represent sub-Gaussian tails ($\beta > 2$) whereas the t cannot. The second is that, while both can have heavy-tails, the generalized normal begins to form a cusp as the origin (as $\beta \to 0$) whereas the t remains smooth.

3. **Mixture of Normals**: Lastly, we consider a mixture of normal distributions:

$$p_u(u) = \sum_{k=1}^{K} \pi_k \, \mathrm{N}(u; \mu_k, \sigma_k)$$

   where k is the component index, $\pi_k \in [0, 1]$ is the weight of the kth component, and (μ_k, σ_k) are the parameters of the kth component. This choice was motivated by previous work that successfully used mixtures to model heavy-tailed distributions [Feldmann and Whitt, 1998, Okada et al., 2020]. There is also theoretical justification: while the components themselves do not have heavy-tails, the mixture can approximate any smooth density arbitrarily well (but perhaps requiring an exponentially large number of components). Lastly, Hagemann and Neumayer [2021] showed that NF training can be stabilized in cases (e.g. discrepancy in support between base and target distributions) by using a mixture for $p_u(\boldsymbol{u})$. We suspect that training on heavy-tailed data will introduce optimization difficulties that may only be exacerbated by having a base density with wide-ranging tail behavior (like the generalized normal). The mixture's tail will adapt much more gradually, possibly improving the stability of gradient descent.

We do not consider multivariate versions of these distributions, instead always assuming they factorize across dimensions. We also do not consider mixtures of t- or generalized normal distributions due to it being difficult to control the tail behavior. For such a mixture, the ν or β parameters might favor light tails for the components, but the overall density could still have heavy tails. For the mixture of normals, we know that tail behavior will be directly controlled by the number of components and their dispersion. The choice of divergence function used for optimization (e.g. forward vs reverse Kullback–Leibler divergence (KLD)) will also have an effect on the model's ability to capture heavy-tailed data. In the experiments, we exclusively use the KLD that corresponds to maximum likelihood estimation: $\mathbb{KL}[p^*(\boldsymbol{x})||p_\phi(\boldsymbol{x})]$, where p^* represents the true density and p_ϕ the NF. This KLD formulation should per-

form the best for heavy tails since the model should be penalized for *underestimating* the support of p^*.

4 EXPERIMENTS

We test the three base densities (Student's t, generalized normal, mixture of normals) in two distinct settings: two-dimensional density estimation and generation of tabular data. As a baseline in both settings, we use a NF whose base is a multivariate normal distribution.

4.1 NEAL'S FUNNEL

Data For our first experiment, we consider a two-dimensional simulation. We follow Jaini et al. [2020], setting the target distribution to be a bi-variate Neal's funnel:

$$\mathbf{x}_i = \begin{cases} x_{i,1} \sim \mathrm{N}(\gamma, 1), \\ x_{i,2} \sim \mathrm{N}(0,\ \exp\{x_{i,1}/2\}). \end{cases}$$

In a slight deviation from Jaini et al. [2020]'s setup, we modify Neal's funnel by adding a log-normal prior over the variance of the second variable via a parameter $\gamma \in \mathbb{R}^{\geq 0}$, which further controls the tail behavior. Density plots for $\gamma \in \{0, 2, 4\}$ are shown in the first column of Figure 1. The funnel is stretched to the left as γ grows.

Model We train an RNVP flow [Dinh et al., 2017] with a depth of 16 using the four aforementioned base distributions. We use the Adam optimizer with a learning rate of 10^{-4}. We train the model for 10^4 iterations with a batch size of 1024 samples.

Results Figure 1 shows 20,000 samples from each NF, colored according to their density under the NF. Figure 2 contains the quantile plots for each NF, choice of base, and target distribution. For the standard Neal's funnel ($\gamma = 0$), all base distributions perform well, even the multivariate normal. Yet in the heavier-tailed cases ($\gamma = 2$ and $\gamma = 4$), the multivariate normal obviously fails. The generalized normal performs well for $\gamma = 2$, but we encountered optimization pathologies for $\gamma = 4$. The Student's t, on the other hand, remains stable but was unable to capture the tails, as is made clear in the quantile plots. Lastly, we found the mixture to perform the best, remaining stable and matching the density well even for $\gamma = 4$. We conjecture that the mixture's performance is partially due to the flexibility of the base and partially due to optimization stability. Regarding the former, the quantile plots show that the mixture is reasonably close to the target in all cases. The Student's t base is similarly close, but does not enjoy the same performance. Thus, we suspect the t's failure might be due to optimization, not expressivity of the base.

4.2 SYNTHETIC DATA GENERATION

Data and Model We next apply the NFs to a real-world tabular data set: *Credit Fraud*[1]. This dataset contains transactions as features and a label identifying each as fraud or legitimate. The dataset is quite challenging: it has extreme class imbalance—284,315 legitimate transactions, 492 fraudulent ones—and contains heavy-tailed features. We again train an RNVP NF with the same architecture as the Neal's funnel experiment but with a depth of 12 due to memory limitations. The rest of the parameters are set as our previous experiment.

Evaluation To evaluate the models, we test whether the synthetic data can be distinguished from the real data. We formulate this task as a supervised learning problem. Synthetic and real data are labeled as such, and a classifier is trained on the combined data with the goal of predicting if an observation is real or synthetic. We use a logistic regressor and a support vector machine as the classifiers. We generated $20{,}000$ samples from the flow and combined them with a stratified sample of $20{,}000$ observations from the actual credit dataset. We use 5-fold cross validation and report the average area under the ROC curve.

Results Table 1 reports the AU-ROC curve results for the two classifiers and for the two data classes (legitimate vs fraudulent). Legitimate transactions comprise 99.8% of the total records, and the multivariate normal and mixture of normals performed best. We conjecture that this was due to the Student's t and generalized normal having too heavy of tails, resulting in their data being easily distinguished. For fraudulent transactions, again the Student's t and generalized normal performed poorly, possibly due to optimization difficulties. The mixture of normals again performed well, leading us to conclude the model was able to adapt to both light- and heavy-tailed situations.

5 CONCLUSIONS AND FUTURE WORK

We empirically studied the effect of the choice of base distribution on a NF's ability to model heavy-tailed data. We found that the mixture of normals, while naive, afforded stable training and seemed to adapt well to both heavy- and light-tailed settings—as demonstrated on a dataset of fraudulent vs legitimate transactions.

In future work, we wish to further understand why the mixture model performs best—whether that be due to optimization or expressivity. We will also perform experiments on additional datasets. Lastly, we also will look to imbue tail-index-specific information into the NF, to make it easier to capture nuanced tail behavior.

[1] https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud

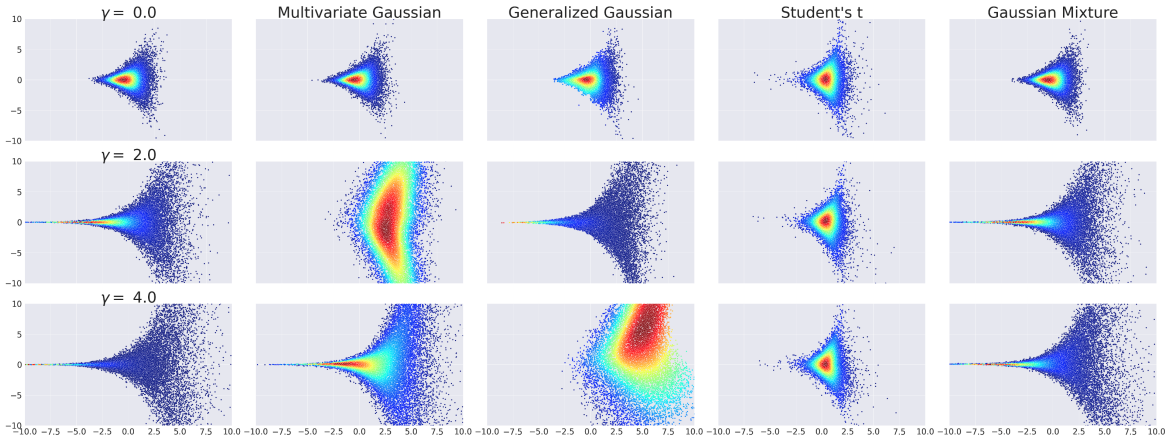


Figure 1: Density plots for RNVP with different bases. We observe that for the heavier-tailed cases the mixture bases performs consistently well while the rest fail in some or all cases.

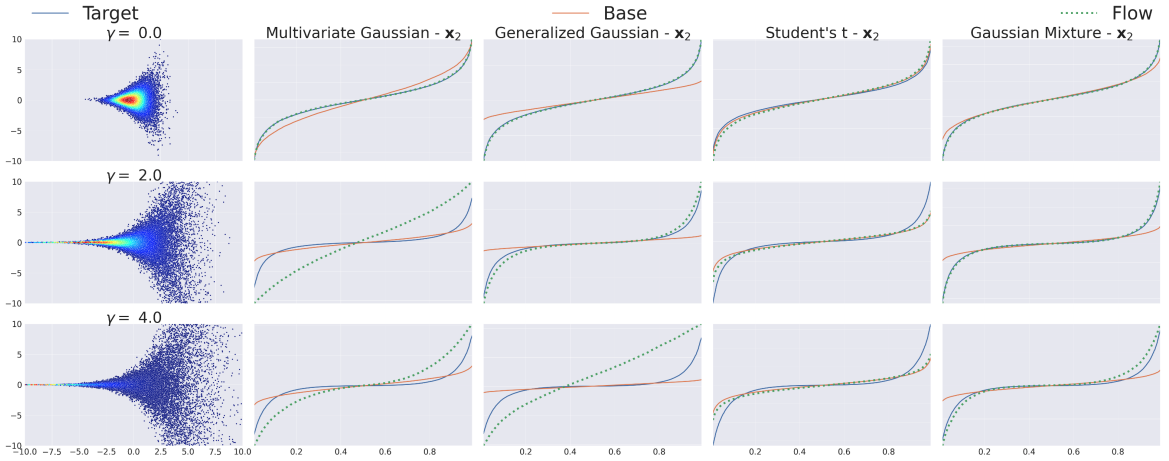


Figure 2: Quantile plots for RNVP with different bases for the second variable of the Neal's funnel $\mathbf{x}_2$. We illustrate the inability of non-mixture bases in capturing the tail behavior of the target distribution for the heavier-tailed cases.

	Area Under ROC Curve (%)			
	Legitimate		Fraudulent	
Base Distribution	Logistic Regression	Support Vector Machine	Logistic Regression	Support Vector Machine
Multivariate Normal	89	**87**	57	59
Student's t	100	100	100	100
Generalized Normal	100	100	100	100
Mixture of Normals	**66**	98	**30**	**41**

Table 1: *Identifying Real vs Synthetic Data*. The table reports the area under the ROC curve for two classifiers, a logistic regressor and a support vector machine. Thus, lower scores are better since it means the classifier had difficulty distinguishing real from synthetic. The mixture performed the best in three out of four cases.

Acknowledgements

This research has been performed as part of the *Enabling Personalized Intervention* (EPI) project. The EPI project is funded by the Dutch Science Foundation in the Commit2Data program, grant number 628.011.028.

References

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *International Conference on Learning Representations*, 2017.

Anja Feldmann and Ward Whitt. Fitting mixtures of exponentials to long-tail distributions to analyze network performance models. *Performance evaluation*, 31(3-4): 245–279, 1998.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Neural Information Processing Systems (NIPS)*, 2014.

Paul Hagemann and Sebastian Neumayer. Stabilizing invertible neural networks using mixture models. *Inverse Problems*, 37(8):085002, 2021.

Jonathan B Hill. On tail index estimation for dependent, heterogeneous data. *Econometric Theory*, 26(5):1398–1436, 2010.

Priyank Jaini, Ivan Kobyzev, Yaoliang Yu, and Marcus Brubaker. Tails of lipschitz triangular flows. In *International Conference on Machine Learning*, pages 4673–4681. PMLR, 2020.

Sanket Kamthe, Samuel Assefa, and Marc Deisenroth. Copula flows for synthetic data generation. *arXiv preprint arXiv:2101.00598*, 2021.

Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in Neural Information Processing Systems*, 31, 2018.

Makoto Okada, Kenji Yamanishi, and Naoki Masuda. Long-tailed distributions of inter-event times as mixtures of exponential distributions. *Royal Society open science*, 7 (2):191643, 2020.

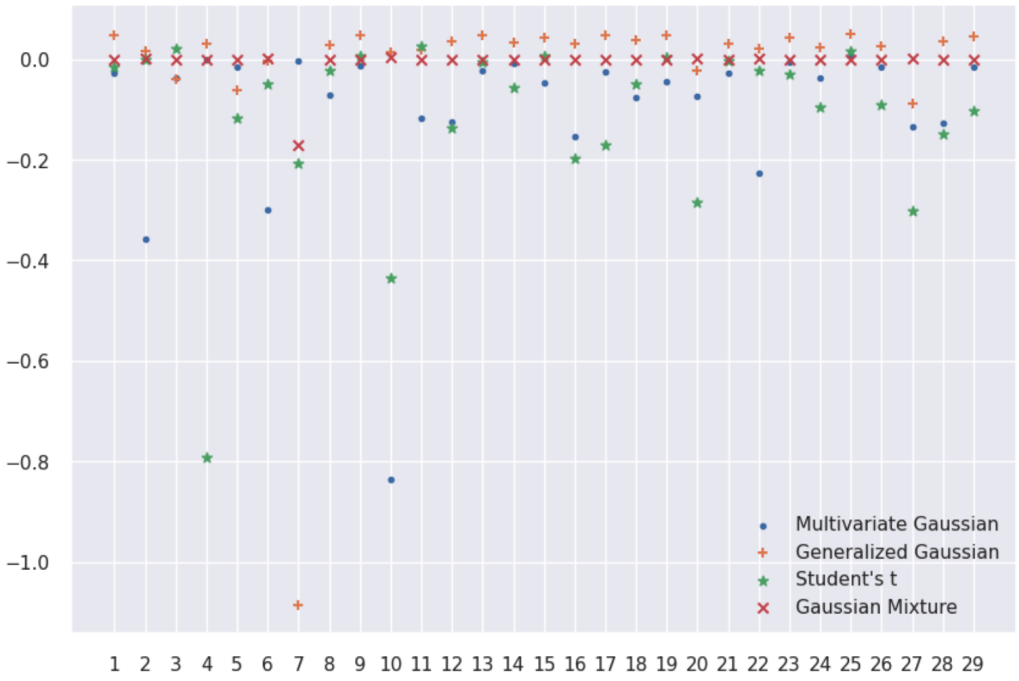
George Papamakarios, Eric Nalisnick, Danilo Jimenez Rezende, Shakir Mohamed, and Balaji Lakshminarayanan. Normalizing flows for probabilistic modeling and inference. *Journal of Machine Learning Research*, 22(57):1–64, 2021. URL `http://jmlr.org/papers/v22/19-1028.html`.

Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

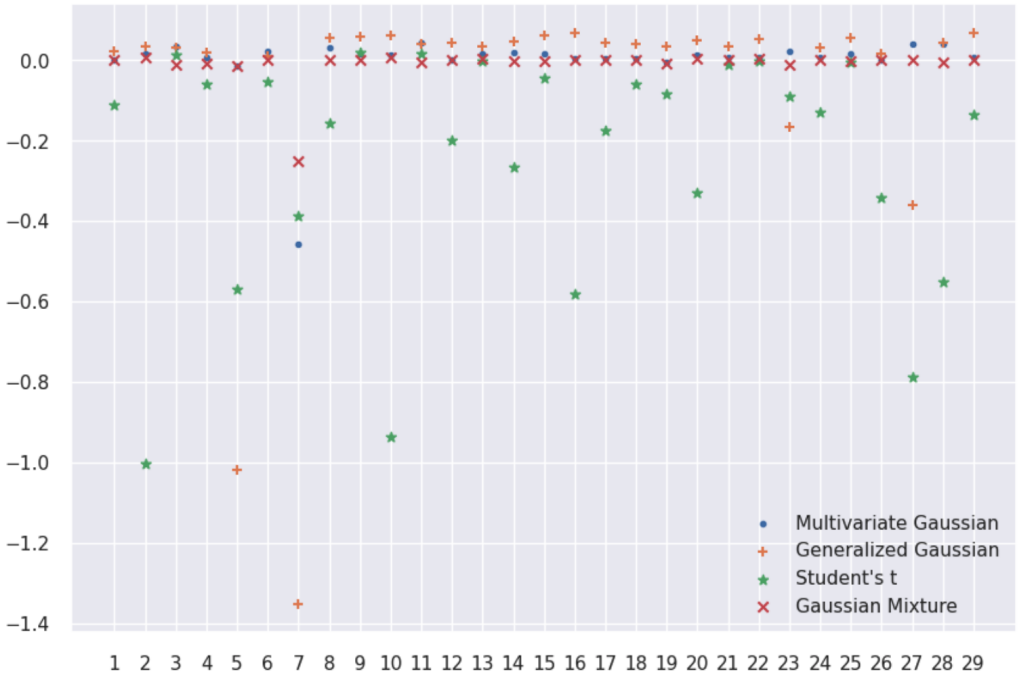
Donald B Rubin. Statistical disclosure limitation. *Journal of official Statistics*, 9(2):461–468, 1993.

Esteban G Tabak and Cristina V Turner. A family of nonparametric density estimation algorithms. *Communications on Pure and Applied Mathematics*, 66(2):145–164, 2013.

A APPENDIX



(a) "Legitimate transactions" class



(b) "Fraudulent transactions" class

Figure 3: Delta of real and estimated tail indices [Hill, 2010] for each feature - tail index is an indicator of how long or short tailed the distribution of a certain random variable is. Negative difference means underestimation of the tail length while positive difference means overestimation. We observe that while the tail with mixture base performs best in capturing the tail behavior of features, other examined base distributions fail to properly capture the tail behavior to various degrees. It is especially apparent for the case of Studentś t distribution with underestimation of most of the tail indices