Variational Autoencoder with Embedded Student-t Mixture Model for Authorship Attribution

Benedikt Boenninghoff,[1] **Steffen Zeiler,**[1] **Robert M. Nickel,**[2] **Dorothea Kolossa**[1]
[1]*Cognitive Signal Processing Group, Ruhr University Bochum, Germany*
[2]*Department of Electrical and Computer Engineering, Bucknell University, Lewisburg, PA, USA*
{benedikt.boenninghoff, dorothea.kolossa, steffen.zeiler}@rub.de
rmn009@bucknell.edu

Abstract

Traditional computational authorship attribution describes a classification task in a closed-set scenario. Given a finite set of candidate authors and corresponding labeled texts, the objective is to determine which of the authors has written another set of anonymous or disputed texts. In this work, we propose a probabilistic autoencoding framework to deal with this supervised classification task. Variational autoencoders (VAEs) have had tremendous success in learning latent representations. However, existing VAEs are currently still bound by limitations imposed by the assumed Gaussianity of the underlying probability distributions in the latent space. In this work, we are extending a VAE with an embedded Gaussian mixture model to a Student-t mixture model, which allows for an independent control of the "heaviness" of the respective tails of the implied probability densities. Experiments over an Amazon review dataset indicate superior performance of the proposed method.

1 Introduction

Supervised authorship attribution traditionally refers to the task of analyzing the linguistic patterns of a text in order to determine who, from a finite set of enrolled authors, has written a document of unknown authorship. Nowadays, the focus of this *closed-set* scenario has shifted from *literary* to *social media* authorship attribution, where methods have been developed to deal with large-scale datasets of small-sized online texts. Examples are provided by the work of (Rocha et al., 2017), (Boenninghoff et al., 2019a), (Theophilo et al., 2019), and (Tschuggnall et al., 2019).

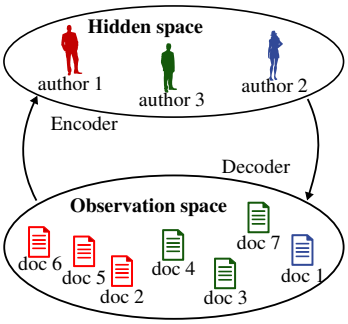


Figure 1: Non-linear observation model for authorship analysis.

The ADHOMINEM system proposed by (Boenninghoff et al., 2019b) is a linguistically motivated deep learning topology that can be seen as a feature extractor for such a tasks. The original ADHOMINEM system is trained on a large dataset of Amazon reviews[1], each written by one of $784,649$ distinct authors. The key aspect of ADHOMINEM is that it is *not* designed to recognize authors but, instead, to recognize a *difference in authorship* between two given texts. As such, the system produces internal *neural* features, i.e. *observation space* features (see Fig. 1), that do not directly represent authorship but rather the stylistic characteristics that distinguish authorship. Since ADHOMINEM is trained with a large number of authors and since the space of writing style variations across authors is generally almost as big as the space of writing style variations itself, it can be argued that features produced by ADHOMINEM can serve as a proxy representation for writing style variations in general. The features produced by ADHOMINEM, therefore, have favorable properties for a variety of writing-style-based classification tasks, including the supervised authorship attribution pursued in this paper.

[1]Each review text had a length of less than 1000 tokens, i.e. relatively short text sizes.

Proceedings of the 28th International Conference on Computational Linguistics, pages 519–529
Barcelona, Spain (Online), December 8-13, 2020

We are proposing the use of a Variational Autoencoder (VAE) framework to identify the authorship of a text based on its ADHOMINEM features. The VAE maps ADHOMINEM features into a *latent space* in which mapped features cluster when they stem from texts written by the same author. In a VAE framework, a probabilistic model is fitted in the latent space, which, in turn, can be exploited for the targeted classification task. In our case, we show that the application of a Student-t Mixture Model (SMM) in the latent space leads to a performance that is superior to the application of the commonly employed Gaussian Mixture Models (GMM).

The conventional VAE framework, as published by (Kingma and Welling, 2013) for example, combines unsupervised deep learning with variational Bayesian methods. A VAE relies on a probabilistic graphical model in the form of a directed acyclic graph, in which the hidden representations of an encoder network as well as the reconstructed outputs of a subsequent decoder network are treated as random variables. More precisely, the encoder defines a variational inference network, using high-dimensional observations to estimate an approximate posterior distribution in the latent space, and the decoder itself is a generative network, mapping latent representations back to distributions over the observation space. The framework is used to generate compressed, approximate representations for virtually any type of patterned input. Depending on the targeted application, we may remove either the encoder or the decoder from the framework, once the joint training of the combined encoder-decoder system has been completed. A VAE can be understood as a single-class probabilistic autoencoder, since it is assumed that all latent representations are sampled from the same Gaussian distribution. Different extensions of the conventional VAE, e.g. (Sohn et al., 2015), (Dilokthanakul et al., 2016), (Nalisnick et al., 2016), (Sønderby et al., 2016), (Johnson et al., 2016), (Nalisnick and Smyth, 2017), (Ebbers et al., 2017), (Lin et al., 2018), (Takahashi et al., 2018), (Davidson et al., 2018), (Domke and Sheldon, 2018), and (Abiri and Ohlsson, 2019), have been proposed. Particularly relevant to our work is the paper by (Jiang et al., 2017), in which the authors broadened the conventional VAE concept by generalizing the assumption of strictly Gaussian distributions to mixtures of Gaussians. This structure represents our baseline in the following.

The advantage of using the Student-t model is that we obtain a means to independently control the *heaviness* of the respective tails of each distribution. Our generalization of the framework can be successfully employed in a variety of common machine learning tasks:

- *Unsupervised learning:* The basic architecture of our proposed method provides a generic recipe to autonomously group high-dimensional data into meaningful clusters.
- *Supervised learning:* The derived loss function of our training method carries a cross-entropy term, which can be used to directly fuse class label information into the learning task. We are thereby able to enforce learning in a predefined/supervised direction as well.
- *Semi-supervised learning:* In some cases, we may have a large amount of training data, only a small subset of which is labeled. In this situation, we can utilize our method to, first, pre-train the model in a supervised manner and then refine the model with the unlabeled data in an unsupervised fashion.

2 Model Description

2.1 Preliminaries

Let $\mathcal{O} = \{\boldsymbol{o}_n\}_{n=1}^N = \{\boldsymbol{o}_1, \ldots, \boldsymbol{o}_N\}$ denote a training set of observation vectors $\boldsymbol{o}_n \in \mathbb{R}^L$ for $n \in \{1 \ldots N\}$. We assume that the $\boldsymbol{o}_n$ are independent and identically distributed samples from either a continuous or a discrete random variable. Furthermore, we use $\mathcal{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ to denote a collection of N low-dimensional latent representation vectors $\boldsymbol{x}_n \in \mathbb{R}^D$, where each $\boldsymbol{x}_n$ is associated with a corresponding observation $\boldsymbol{o}_n$. Following (Murphy, 2012), we define the Student-t distribution for the n-th latent representation $\boldsymbol{x}_n$ by assuming that this D-dimensional vector belongs to the k-th cluster with $k \in \{1, \ldots K\}$ as

$$\mathcal{S}(\boldsymbol{x}_n | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \nu_k) = \frac{\Gamma(\frac{\nu_k + D}{2})}{\Gamma(\frac{\nu_k}{2})} \frac{\det(\boldsymbol{\Sigma}_k)^{-\frac{1}{2}}}{(\pi\nu_k)^{\frac{D}{2}}} \left[1 + \tfrac{1}{\nu_k}(\boldsymbol{x}_n - \boldsymbol{\mu}_k)^T \boldsymbol{\Sigma}_k^{-1} (\boldsymbol{x}_n - \boldsymbol{\mu}_k)\right]^{-\frac{\nu_k + D}{2}}$$

where $\boldsymbol{\mu}_k$ defines the D-dimensional mean vector of the k-th class, $\boldsymbol{\Sigma}_k$ denotes the $D \times D$ scale matrix and $\nu_k \in [0, \infty]$ is the number of degrees of freedom. For $\nu_k \to \infty$, the Student-t distribution tends towards a Gaussian distribution of the same mean vector and covariance matrix. Alternatively, we can understand the Student-t distribution as a marginalization with respect to a hidden variable, i.e.

$$\mathcal{S}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \nu_k) = \int_0^{+\infty} \mathcal{N}\big(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k/u_{nk}\big)\, \mathcal{G}\big(u_{nk}|\tfrac{\nu_k}{2}, \tfrac{\nu_k}{2}\big)\, \mathrm{d}u_{nk} \tag{1}$$

where $u_{nk} > 0$ is the hidden scale variable. The normal distribution is defined as

$$\mathcal{N}\big(\boldsymbol{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}/u\big) = \tfrac{1}{\sqrt{\det(2\pi\boldsymbol{\Sigma}/u)}} \exp\big\{-\tfrac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\big(\boldsymbol{\Sigma}/u\big)^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\big\}. \tag{2}$$

The term $\mathcal{G}(\cdot)$ is the Gamma distribution given in the form

$$\mathcal{G}\big(u|\alpha, \beta\big) = \frac{\beta^\alpha}{\Gamma(\alpha)}\, u^{\alpha-1} \exp\big\{-\beta u\big\} \tag{3}$$

for $u > 0$ and $\alpha, \beta > 0$. A finite SMM is defined as a weighted sum of multivariate Student-t distributions. With $\sum_{k=1}^K \pi_k = 1$ we may write $p(\boldsymbol{x}_n) = \sum_{k=1}^K \Pr(z_{nk}=1)\; p(\boldsymbol{x}_n|z_{nk}=1) = \sum_{k=1}^K \pi_k\, \mathcal{S}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \nu_k)$. As mentioned by (Svensén and Bishop, 2005) and (Archambeau and Verleysen, 2007), we can view the Student-t distribution in Eq. (1) as the marginalization of a Gaussian-Gamma distribution by integrating out the hidden scale variable u_{nk}. This infinte mixture of normal distributions with the same mean vector but with varying covariance matrices can be incorporated into a generative process. Omitting the dependency on the hyper-parameters $\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k$ and ν_k, Eq. (1) can be rewritten as $\mathcal{S}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k, \nu_k) = p(\boldsymbol{x}_n|z_{nk}=1) = \int_0^{+\infty} p(u_{nk}|z_{nk}=1)\; p(\boldsymbol{x}_n|u_{nk}, z_{nk}=1)\; \mathrm{d}u_{nk}$, where $z_{nk} \in \{0, 1\}$ is an indicator variable showing whether the n-th observation belongs to the k-th class. Consequently, our generative model in the latent space is augmented by the scale parameter u_{nk} as an additional latent variable.

2.2 The Generative Model

We use $\boldsymbol{\xi} = \{\boldsymbol{\mu}, \boldsymbol{\nu}, \boldsymbol{\Sigma}, \boldsymbol{\pi}\} = \{\boldsymbol{\mu}_k, \nu_k, \boldsymbol{\Sigma}_k, \pi_k\}_{k=1}^K$ to denote the set of all hyper-parameters of an SMM. We can generate observation samples $\boldsymbol{o}_n$ for the proposed Student-t VAE with the following 5 steps:

1. Choose a cluster for the n-th observation by sampling the one-hot vector $\boldsymbol{z}_n \sim p(\boldsymbol{z}_n)$, where
$$p_{\boldsymbol{\xi}}(\boldsymbol{z}_n) = \prod_{k=1}^K \pi_k^{z_{nk}} \tag{4}$$
and $\boldsymbol{z}_n = \big[z_{n1}, \ldots, z_{nK}\big]^T$.
2. Sample the n-th scale vector $\boldsymbol{u}_n \sim p_{\boldsymbol{\xi}}(\boldsymbol{u}_n|\boldsymbol{z}_n)$, where
$$p_{\boldsymbol{\xi}}(\boldsymbol{u}_n|\boldsymbol{z}_n) = \prod_{k=1}^K \mathcal{G}\big(u_{nk}|\tfrac{\nu_k}{2}, \tfrac{\nu_k}{2}\big)^{z_{nk}} \tag{5}$$
and $\boldsymbol{u}_n = \big[u_{n1}, \ldots, u_{nK}\big]^T$.
3. Sample a new latent representation for the n-th observation, $\boldsymbol{x}_n \sim p_{\boldsymbol{\xi}}(\boldsymbol{x}_n|\boldsymbol{z}_n, \boldsymbol{u}_n)$ with
$$p_{\boldsymbol{\xi}}(\boldsymbol{x}_n|\boldsymbol{z}_n, \boldsymbol{u}_n) = \prod_{k=1}^K \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k/u_{nk})^{z_{nk}}. \tag{6}$$
4. Decode a parameter set for the n-th observation $\boldsymbol{o}_n$,
$$\big\{\boldsymbol{\mu}_n^{(\boldsymbol{o}|\boldsymbol{x})}, \log \boldsymbol{\sigma}_n^{(\boldsymbol{o}|\boldsymbol{x})}\big\} = \text{Decoder}_{\boldsymbol{\theta}}(\boldsymbol{x}_n). \tag{7}$$
The set $\boldsymbol{\theta}$ summarizes all weights and bias terms of the decoder network.
5. Sample an observation $\boldsymbol{o}_n \sim p_{\boldsymbol{\theta}}(\boldsymbol{o}_n|\boldsymbol{x}_n)$, where
$$p_{\boldsymbol{\theta}}(\boldsymbol{o}_n|\boldsymbol{x}_n) = \mathcal{N}\big(\boldsymbol{o}_n|\boldsymbol{\mu}_n^{(\boldsymbol{o}|\boldsymbol{x})}, \boldsymbol{\Sigma}_n^{(\boldsymbol{o}|\boldsymbol{x})}\big) \tag{8}$$
with $\boldsymbol{\Sigma}_n^{(\boldsymbol{o}|\boldsymbol{x})} = \text{diag}\big\{\big(\boldsymbol{\sigma}_n^{(\boldsymbol{o}|\boldsymbol{x})}\big)^2\big\}$.

The generative process for the proposed Student-t VAE as a graphical model is illustrated in Fig. 2.

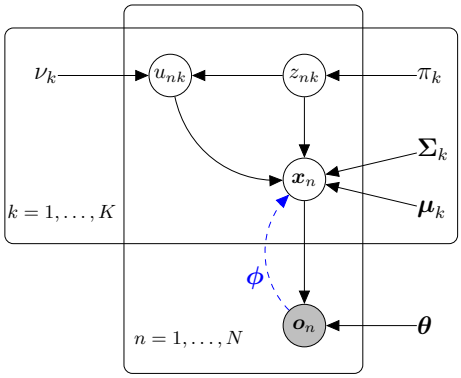


Figure 2: Probabilistic graphical model for the generative process of the proposed Student-t VAE.

2.3 Approximate Inference

At this point it is notationally beneficial to define the set $\mathcal{H} = \{\mathcal{Z}, \mathcal{U}, \mathcal{X}\} = \{\{z_{nk}, u_{nk}\}_{k=1}^{K}, \boldsymbol{x}_n\}_{n=1}^{N}$ of all latent variables of our proposed framework. We apply the mean-field approximation to find an analytical expression of the approximate joint posterior distribution $q_{\boldsymbol{\phi}}(\mathcal{H})$. The symbol $\boldsymbol{\phi}$ is used to represent the set of all weights and bias terms of the underlying encoder network. Suppose, the joint posterior distribution of $\mathcal{H}$ can be factored such that $q_{\boldsymbol{\phi}}(\mathcal{H}|\mathcal{O}) = \prod_i q_{\boldsymbol{\phi}}(\mathcal{H}_i|\mathcal{O})$, then the posterior distribution can be obtained according to (Bishop, 2006) from: $\ln q_{\boldsymbol{\phi}}(\mathcal{H}_j|\mathcal{O}) = \mathbb{E}_{\prod_{i \neq j} q_{\boldsymbol{\phi}}(\mathcal{H}_i|\mathcal{O})}\big[\ln p(\mathcal{O}, \mathcal{H})\big] + \text{const}$. In our context, the product $\prod_i q(\mathcal{H}_i|\mathcal{O})$ represents a suitable factorization of the joint posterior distribution of all latent variables. One possible approximate factorization is:

$$q_{\boldsymbol{\phi}}(\mathcal{H}|\mathcal{O}) \approx \prod_{n=1}^{N} q_{\boldsymbol{\phi}}(\boldsymbol{x}_n|\boldsymbol{o}_n) \prod_{k=1}^{K} q(z_{nk}, u_{nk}). \tag{9}$$

The employed generative model implies that there is a statistical dependency between $\boldsymbol{x}_n$ and $\boldsymbol{z}_n, \boldsymbol{u}_n$. It can be argued, however, that we may ignore this dependency in our case because the posterior distribution in the latent space is encoded by the second neural network, i.e. $q_{\boldsymbol{\phi}}(\boldsymbol{x}_n|\boldsymbol{o}_n) = \mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_n^{(\boldsymbol{x}|\boldsymbol{o})}, \boldsymbol{\Sigma}_n^{(\boldsymbol{x}|\boldsymbol{o})})$ with $\{\boldsymbol{\mu}_n^{(\boldsymbol{x}|\boldsymbol{o})}, \log \boldsymbol{\sigma}_n^{(\boldsymbol{x}|\boldsymbol{o})}\} = \text{Encoder}_{\boldsymbol{\phi}}(\boldsymbol{o}_n)$, where, again, $\boldsymbol{\Sigma}_n^{(\boldsymbol{x}|\boldsymbol{o})} = \text{diag}\big\{\big(\boldsymbol{\sigma}_n^{(\boldsymbol{x}|\boldsymbol{o})}\big)^2\big\}$. Note that the posterior distribution of $\boldsymbol{z}_n$ and $\boldsymbol{u}_n$ in Eq. (9) does not directly depend on $\boldsymbol{\phi}$, which is important for the calculation of the loss function discussed in Section 2.4. It is not necessary to approximate the joint posterior distribution of $\boldsymbol{z}_n$ and $\boldsymbol{u}_n$, as it is possible to analytically determine the marginal distributions $q(z_{nk})$ and $q(u_{nk}|z_{nk})$ given the joint distribution $q(z_{nk}, u_{nk})$. We can apply the mean-field approximation to compute the joint distribution $q(z_{nk} = 1, u_{nk})$. The marginal distribution can be derived via $q(z_{nk} = 1) = \int_0^{+\infty} q(z_{nk} = 1, u_{nk})\, \mathrm{d}u_{nk}$. We can now determine the posterior distribution $q(u_{nk}|z_{nk} = 1)$ as:

$$q(u_{nk}|z_{nk} = 1) = \frac{q(u_{nk}, z_{nk} = 1)}{q(z_{nk} = 1)} = \mathcal{G}\big(u_{nk}|\alpha_k, \beta_{nk}\big), \tag{10}$$

where $\alpha_k = \frac{\nu_k + D}{2}$ and $\beta_{nk} = \frac{1}{2}\big[\nu_k + \text{Tr}\big\{\boldsymbol{\Sigma}_n^{(\boldsymbol{x}|\boldsymbol{o})}\boldsymbol{\Sigma}_k^{-1}\big\} + (\boldsymbol{\mu}_n^{(\boldsymbol{x}|\boldsymbol{o})} - \boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}_k^{-1}(\boldsymbol{\mu}_n^{(\boldsymbol{x}|\boldsymbol{o})} - \boldsymbol{\mu}_k)\big]$ define the hyper-parameters for $q(u_{nk}|z_{nk} = 1)$.

2.4 The Variational Lower Bound and the Loss Function

Following (Kingma and Welling, 2013), the negative of the derived evidence lower bound provides a loss function, i.e.

$$J_{\boldsymbol{\phi},\boldsymbol{\theta},\boldsymbol{\xi}}(\mathcal{O}) = -\frac{1}{N}\sum_{n=1}^{N} \mathbb{E}_{q_{\boldsymbol{\phi}}(\mathcal{X},\mathcal{Z},\mathcal{U}|\mathcal{O})}\left[\ln\left\{\frac{p_{\boldsymbol{\theta},\boldsymbol{\xi}}(\mathcal{Z},\mathcal{U},\mathcal{X},\mathcal{O})}{q_{\boldsymbol{\phi}}(\mathcal{X},\mathcal{Z},\mathcal{U}|\mathcal{O})}\right\}\right] = -\frac{1}{N}\sum_{n=1}^{N}\mathcal{L}_{\boldsymbol{\phi},\boldsymbol{\theta},\boldsymbol{\xi}}(\boldsymbol{o}_n) \tag{11}$$

The lower bound for the n-th observation can, thus, be partitioned into the 6 terms:

$$\mathcal{L}_{\boldsymbol{\phi},\boldsymbol{\theta},\boldsymbol{\xi}}(\boldsymbol{o}_n) = \mathbb{E}_{q(\boldsymbol{z}_n)}\big[\ln p_{\boldsymbol{\xi}}(\boldsymbol{z}_n)\big] \tag{12}$$

$$+ \mathbb{E}_{q(\boldsymbol{u}_n,\boldsymbol{z}_n)}\big[\ln p_{\boldsymbol{\xi}}(\boldsymbol{u}_n|\boldsymbol{z}_n)\big] \tag{13}$$

$$+ \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{x}_n|\boldsymbol{o}_n)q(\boldsymbol{u}_n,\boldsymbol{z}_n)}\big[\ln p_{\boldsymbol{\xi}}(\boldsymbol{x}_n|\boldsymbol{z}_n,\boldsymbol{u}_n)\big] \tag{14}$$

$$+ \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{x}_n|\boldsymbol{o}_n)}\big[\ln p_{\boldsymbol{\theta}}(\boldsymbol{o}_n|\boldsymbol{x}_n)\big] \tag{15}$$

$$- \mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{x}_n|\boldsymbol{o}_n)}\big[\ln q_{\boldsymbol{\phi}}(\boldsymbol{x}_n|\boldsymbol{o}_n)\big] \tag{16}$$

$$- \mathbb{E}_{q(\boldsymbol{u}_n,\boldsymbol{z}_n)}\big[\ln q(\boldsymbol{u}_n,\boldsymbol{z}_n)\big]. \tag{17}$$

For the sake of clarity we will discuss each term of the above expression separately. First, we may note that Term (17) remains constant during the gradient-based update phase, i.e. we have $\mathbb{E}_{q(\boldsymbol{u}_n,\boldsymbol{z}_n)}\big[\ln q(\boldsymbol{u}_n,\boldsymbol{z}_n)\big] = \text{constant}$, since there is no dependency on the update parameters in $\boldsymbol{\theta}$, $\boldsymbol{\phi}$ and $\boldsymbol{\xi}$. By assuming ergodicy, we can make the following approximation for Term (15):

$$\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{x}_n|\boldsymbol{o}_n)}\big[\ln p(\boldsymbol{o}_n|\boldsymbol{x}_n)\big] \approx \frac{1}{T}\sum_{t=1}^{T}\ln\mathcal{N}(\boldsymbol{o}_n|\boldsymbol{\mu}_{n,t}^{(\boldsymbol{o}|\boldsymbol{x})},\boldsymbol{\Sigma}_{n,t}^{(\boldsymbol{o}|\boldsymbol{x})}). \tag{18}$$

For the re-parameterization trick, $\boldsymbol{x}_{n,t}$ is obtained as follows: $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\mathbf{0}_{D\times 1},\boldsymbol{I}_{D\times D})$ and $\boldsymbol{x}_{n,t} = \boldsymbol{\mu}_n^{(\boldsymbol{x}|\boldsymbol{o})} + \boldsymbol{\sigma}_n^{(\boldsymbol{x}|\boldsymbol{o})}\odot\boldsymbol{\epsilon}_t$, which is fed into the decoder: $\{\boldsymbol{\mu}_{n,t}^{(\boldsymbol{o}|\boldsymbol{x})},\log\boldsymbol{\sigma}_{n,t}^{(\boldsymbol{o}|\boldsymbol{x})}\} = \text{Decoder}_{\boldsymbol{\theta}}\big(\boldsymbol{x}_{n,t}\big)$. Considering Term (16) we may exploit the entropy of multivariate Gaussian distributions, i.e.

$$-\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{x}_n|\boldsymbol{o}_n)}\big[\ln q_{\boldsymbol{\phi}}(\boldsymbol{x}_n|\boldsymbol{o}_n)\big] = H\big(\mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_n^{(\boldsymbol{x}|\boldsymbol{o})},\boldsymbol{\Sigma}_n^{(\boldsymbol{x}|\boldsymbol{o})})\big). \tag{19}$$

Term (12) can be summarized as

$$\mathbb{E}_{q(\boldsymbol{z}_n)}\big[\ln p_{\boldsymbol{\xi}}(\boldsymbol{z}_n)\big] = \sum_{k=1}^{K}\gamma_{nk}\ln\pi_k, \tag{20}$$

in which γ_{nk} describes the posterior class probabilities such that $q(\boldsymbol{z}_n) = \prod_{k=1}^{K}\gamma_{nk}^{z_{nk}}$ with $\gamma_{nk} = q(z_{nk}=1)/\sum_{k'}q(z_{nk'}=1)$. For Term (13), it follows that

$$\mathbb{E}_{q(\boldsymbol{u}_n,\boldsymbol{z}_n)}\big[\ln p_{\boldsymbol{\xi}}(\boldsymbol{u}_n|\boldsymbol{z}_n)\big] = \sum_{\forall k}\gamma_{nk}\,\Big(\ln(\frac{\nu_k}{2})^{\frac{\nu_k}{2}} - \ln\Gamma(\frac{\nu_k}{2}) + \big(\tfrac{\nu_k}{2}-1\big)\big[\psi(\alpha_k)-\ln\beta_{nk}\big] - \tfrac{\nu_k}{2}\big[\tfrac{\alpha_k}{\beta_{nk}}\big]\Big)$$

where $\psi(\cdot)$ denotes the Digamma function (Bishop, 2006). To conclude, Term (14) can be decomposed as follows:

$$\begin{aligned}
&\mathbb{E}_{q_{\boldsymbol{\phi}}(\boldsymbol{x}_n|\boldsymbol{o}_n)q(\boldsymbol{u}_n,\boldsymbol{z}_n)}\big[\ln p_{\boldsymbol{\xi}}(\boldsymbol{x}_n|\boldsymbol{z}_n,\boldsymbol{u}_n)\big]\\
&= \sum_{k=1}^{K}\gamma_{nk}\Big(-\tfrac{D}{2}\ln\big\{2\pi\big\} - \tfrac{1}{2}\ln\det(\boldsymbol{\Sigma}_k) + \tfrac{D}{2}\big[\psi(\alpha_k)-\ln\beta_{nk}\big]\\
&\qquad - \tfrac{\alpha_k}{2\,\beta nk}\big(\text{Tr}\big\{\boldsymbol{\Sigma}_n^{(\boldsymbol{x}|\boldsymbol{o})}\boldsymbol{\Sigma}_k^{-1}\big\} + (\boldsymbol{\mu}_n^{(\boldsymbol{x}|\boldsymbol{o})}-\boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}_k^{-1}(\boldsymbol{\mu}_n^{(\boldsymbol{x}|\boldsymbol{o})}-\boldsymbol{\mu}_k)\big)\Big).
\end{aligned} \tag{21}$$

2.5 Interpretation of the Lower Bound

As a result, the Evidence Lower Bound (ELBO) for the n-th observation defined through Terms (12) to (17) can be rewritten. We obtain the following, more compact expression:

$$\mathcal{L}_{\boldsymbol{\phi},\boldsymbol{\theta},\boldsymbol{\xi}}\big(\boldsymbol{o}_n\big) = \frac{1}{T}\sum_{t=1}^{T}\ln\mathcal{N}(\boldsymbol{o}_n|\boldsymbol{\mu}_{n,t}^{(\boldsymbol{o}|\boldsymbol{x})},\boldsymbol{\Sigma}_{n,t}^{(\boldsymbol{o}|\boldsymbol{x})}) + H\big(\mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_n^{(\boldsymbol{x}|\boldsymbol{o})},\boldsymbol{\Sigma}_n^{(\boldsymbol{x}|\boldsymbol{o})})\big) + \sum_{k=1}^{K}\gamma_{nk}\ln\rho_{nk}, \tag{22}$$

where $\ln\rho_{nk} = \ln q(z_{nk}=1) - H\big(\mathcal{G}(u_{nk}|\alpha_k,\beta_{nk})\big) - \frac{D}{2}\ln\big\{2\pi\big\}$. $H\big(\mathcal{G}(u_{nk}|\alpha_k,\beta_{nk})\big)$ represents the entropy of the Gamma distribution with parameters α_k and β_{nk}. Similarly to the ELBO of the conventional VAE, we may interpret the function of each term of the derived lower bound in Eq. (22). The first term represents the reconstruction error, measuring how well the encoder-decoder framework fits the dataset. The second term can be seen as a regularizer quantifying the output of the decoder. Following the maximum entropy principle, it will maximize the uncertainty with regard to possibly missing information. The third term, the cross-entropy, evaluates the clustering or classification. In the case of supervised learning, γ_{nk} is replaced by the true class labels. All terms in Eq. (22) can easily be computed batch-wise. The training procedure of the proposed Student-t VAE is summarized in Algorithm 1.

Algorithm 1: Student-t VAE Training Procedure

Initialize $\boldsymbol{\theta}$, $\boldsymbol{\phi}$ and $\boldsymbol{\xi}$
for epoch $= 1, \ldots,$ max_number_epochs **do**
  $J = 0$
  for $n = 1, \ldots, N$ **do** ▷ encoding, decoding
    $\{\boldsymbol{\mu}_n^{(\boldsymbol{x}|\boldsymbol{o})}, \log \boldsymbol{\sigma}_n^{(\boldsymbol{x}|\boldsymbol{o})}\} = \texttt{Encoder}_{\boldsymbol{\phi}}(\boldsymbol{o}_n)$ with $\boldsymbol{\Sigma}_n^{(\boldsymbol{x}|\boldsymbol{o})} = \text{diag}\{(\boldsymbol{\sigma}_n^{(\boldsymbol{x}|\boldsymbol{o})})^2\}$
    for $t = 1, \ldots, T$ **do**
      $\boldsymbol{\epsilon}_t \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$
      $\boldsymbol{x}_{n,t} = \boldsymbol{\mu}_n^{(\boldsymbol{x}|\boldsymbol{o})} + \boldsymbol{\sigma}_n^{(\boldsymbol{x}|\boldsymbol{o})} \odot \boldsymbol{\epsilon}_t$
      $\{\boldsymbol{\mu}_{n,t}^{(\boldsymbol{o}|\boldsymbol{x})}, \log \boldsymbol{\sigma}_{n,t}^{(\boldsymbol{o}|\boldsymbol{x})}\} = \texttt{Decoder}_{\boldsymbol{\theta}}(\boldsymbol{x}_{n,t})$ with $\boldsymbol{\Sigma}_{n,t}^{(\boldsymbol{o}|\boldsymbol{x})} = \text{diag}\{(\boldsymbol{\sigma}_{n,t}^{(\boldsymbol{o}|\boldsymbol{x})})^2\}$
    end for
    for $k = 1, \ldots, K$ **do** ▷ hyper-parameters
      $\alpha_k = \frac{\nu_k + D}{2}$
      $\beta_{nk} = \frac{1}{2}\big[\nu_k + \text{Tr}\{\boldsymbol{\Sigma}_n^{(\boldsymbol{x}|\boldsymbol{o})}\boldsymbol{\Sigma}_k^{-1}\} + (\boldsymbol{\mu}_n^{(\boldsymbol{x}|\boldsymbol{o})} - \boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}_k^{-1}(\boldsymbol{\mu}_n^{(\boldsymbol{x}|\boldsymbol{o})} - \boldsymbol{\mu}_k)\big]$
      $\ln qz_{nk} = \ln \pi_k + \frac{\nu_k}{2}\ln\frac{\nu_k}{2} - \ln\Gamma(\frac{\nu_k}{2}) - \frac{1}{2}\ln\det(\boldsymbol{\Sigma}_k) + \ln\Gamma(\alpha_k) - \alpha_k \ln \beta_{nk}$
      $\ln \rho_{nk} = \ln qz_{nk} - H\big(\mathcal{G}(u_{nk}|\alpha_k, \beta_{nk})\big) - \frac{D}{2}\ln\{2\pi\}$
    end for
    $\ln \boldsymbol{qz}_n = \big[\ln qz_{n1}, \ldots, \ln qz_{nK}\big]^T$
    $\boldsymbol{\gamma}_n = \texttt{Softmax}\big(\ln \boldsymbol{qz}_n\big)$
    $\boldsymbol{\gamma}_n = \big[\gamma_{n1}, \ldots, \gamma_{nK}\big]^T$
    for $k = 1, \ldots, K$ **do** ▷ loss function
      $J -= \gamma_{nk}\ \ln \rho_{nk}$
    end for
    for $t = 1, \ldots, T$ **do**
      $J -= \frac{1}{T}\ln\mathcal{N}(\boldsymbol{o}_n|\boldsymbol{\mu}_{n,t}^{(\boldsymbol{o}|\boldsymbol{x})}, \boldsymbol{\Sigma}_{n,t}^{(\boldsymbol{o}|\boldsymbol{x})})$
    end for
    $J -= H\big(\mathcal{N}(\boldsymbol{x}_n|\boldsymbol{\mu}_n^{(\boldsymbol{x}|\boldsymbol{o})}, \boldsymbol{\Sigma}_n^{(\boldsymbol{x}|\boldsymbol{o})})\big)$
  end for
  $J/ = N$
  Determine and update gradients of parameters in $\boldsymbol{\theta}$, $\boldsymbol{\phi}$, $\boldsymbol{\xi}$
end for

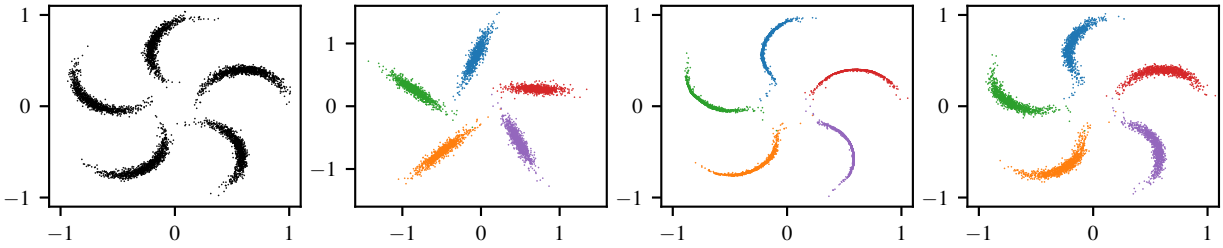
3 Evaluation

The following two sections provide results of two experiments. In the first, we applied the proposed method to purely synthetic data in order to produce a graphical representation that illustrates how the method works in an unsupervised learning scenario. Results are discussed in Section 3.1. In the second experiment, we applied the proposed method to a supervised authorship attribution task and compared the results to a selection of reference methods. Details are discussed in Section 3.2.

We are referring to the proposed SMM-based Variational Autoencoder method, i.e. the Student-t VAE method discussed in Section 2, simply as the tVAE method. As reference methods we have implemented a GMM-based Variational Autoencoder, referred to as gVAE, and two types of Support Vector Machines (SVMs), one linear and one non-linear. The gVAE system was inspired by (Ebbers et al., 2017) and is, in structure, very similar to the method presented by (Jiang et al., 2017). For the sake of a fair comparison, we ensured that the network architecture of both, the tVAE and the gVAE implementations, were exactly the same[2]. Both algorithms are implemented in Python, where the training of the neural networks is accomplished with Tensorflow. The code is available to interested readers upon request.

Our implementations contain the following modifications relative to (Ebbers et al., 2017): It is ensured that the mixing weights sum to one and that each covariance matrix is invertible. Hence, instead of directly updating π_k, $\boldsymbol{\Sigma}_k$ and ν_k, we introduced auxiliary variables such that $\big[\pi_1, \ldots, \pi_k\big]^T = \text{Softmax}\big(\big[m_1, \ldots, m_K\big]^T\big)$, $\nu_k = \log\big(\exp(n_k) + \exp(2.0 + \epsilon)\big)$ and $\boldsymbol{\Sigma}_k = \boldsymbol{C}_k\boldsymbol{C}_k^T + \sigma_k^2\boldsymbol{I}_{D\times D}$ where σ_k^2 is a fixed hyper-parameter. More precisely, we enforced $\nu_k > 2$ by applying a modified softplus-function and we enforced the positive definiteness of $\boldsymbol{\Sigma}_k$ through a Cholesky decomposition by constructing trainable lower-triangular matrices $\boldsymbol{C}_k$ with exponentiated (positive) diagonal elements. We computed and updated the gradients of n_k, m_k and $\boldsymbol{C}_k$ with respect to the loss function.

[2]Except, of course, for the numeric differences in all trainable parameters, i.e. the neural network parameters as well as the mixture model parameters.



(a) Training data points (b) $\boldsymbol{x}_n \sim \mathcal{N}(\boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k/u_{nk})$ (c) mean values $\boldsymbol{\mu}_n^{(\boldsymbol{o}|\boldsymbol{x})}$ (d) $\boldsymbol{o}_n \sim \mathcal{N}\big(\boldsymbol{\mu}_n^{(\boldsymbol{o}|\boldsymbol{x})}, \boldsymbol{\Sigma}_n^{(\boldsymbol{o}|\boldsymbol{x})}\big)$

Figure 3: Results of the Student-t VAE after training with the synthetic pinwheel data set.

3.1 Synthetic Data Experiment for Clustering

To illustrate the inner workings of the tVAE method, we first performed clustering on low-dimensional synthetic spirals of noisy data. The dataset as well as the clustering results are shown in Fig. 3. It is the same dataset used in (Dilokthanakul et al., 2016) and (Johnson et al., 2016). Encoder and decoder are fully connected feed-forward networks (with `ReLU` activation functions) of the form L-H-H-D and D-H-H-L, respectively, where $L = 2$ defines the dimension of the observations, $H = 512$ represents the number of hidden nodes and $D = 2$ is the latent space dimension. We used the Adam optimizer (Kingma and Ba, 2014) to calculate parameter updates. A key problem for the training of VAEs with embedded mixture models is an over-regularization behavior that occurs at the beginning of the training phase. Following (Yeung et al., 2017), it is caused by the regularization term of the ELBO. Both, the prior distribution and the posterior can be decomposed into univariate distributions and therefore we can also decompose the Kullback-Leibler (KL) divergence, $\mathrm{KL}\big(q_{\boldsymbol{\phi}}(\boldsymbol{x}_n|\boldsymbol{o}_n)||p(\boldsymbol{x}_n)\big) = \sum_{d=1}^{D}\mathrm{KL}\big(q_{\boldsymbol{\phi}}(x_{n,d}|\boldsymbol{o}_n)||p(x_{n,d})\big)$, where $x_{n,d}$ is the d-th component of $\boldsymbol{x}_n$. As mentioned in (Yeung et al., 2017), the model has to minimize the KL term *en bloc* and not component-wise. One obvious option for the model is to enforce a large number of components $x_{n,d}$ helping to minimize the KL term, which means these components are (close to) zero. Similarly, our model has to maximize the cross-entropy in Eq. (22), which includes maximizing $\beta_{nk} \approx \mathrm{Tr}\big\{\boldsymbol{\Sigma}_n^{(\boldsymbol{x}|\boldsymbol{o})}\boldsymbol{\Sigma}_k^{-1}\big\} + (\boldsymbol{\mu}_n^{(\boldsymbol{x}|\boldsymbol{o})} - \boldsymbol{\mu}_k)^T\boldsymbol{\Sigma}_k^{-1}(\boldsymbol{\mu}_n^{(\boldsymbol{x}|\boldsymbol{o})} - \boldsymbol{\mu}_k)$ in Algorithm 1. It can happen that the covariances $\boldsymbol{\Sigma}_k$ get smaller and smaller, except for a single global class. This leads to the "anti-clustering behavior" mentioned in (Jiang et al., 2017).

To handle the over-regularization problem, we first trained a GMM to initialize the parameters π_k, $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ of each Student-t mixture component. A similar strategy was suggested in (Dilokthanakul et al., 2016). With a simple modification we can circumvent the tendency of early class merging: If we treat the obtained GMM weights as class labels for the first $10 - 15$ iterations (alternatively, one can randomly assign cluster labels), the neural networks become sufficiently stable and the merging effect is eliminated. The degrees of freedom ν_k were initialized with ≈ 5 for all k. The number of clusters K was known *a priori* and therefore kept fixed for all presented experiments.

After the completion of the training we used the decoder to sample new observations. Fig. 3b illustrates the linearly separable, learned manifolds in latent space by plotting samples drawn from each mixture component according to Eqs. (5) and (6). In Fig. 3c and 3d the mean values and new sampled observations are shown after applying the decoder to the latent data.

3.2 Authorship Attribution

In the previous section, we have shown that the proposed tVAE model has the ability to learn a non-linear generative process. Next, we examine to what extent the tVAE framework can be used to accomplish authorship attribution for Amazon review data.

3.2.1 Feature Extraction

As already discussed in Section 1, we are using the ADHOMINEM system by (Boenninghoff et al., 2019b) as a feature extraction mechanism to convert variable-length documents $\mathcal{D}_n$ (for $n = 1, 2, \ldots$) into fixed-length feature vectors $\boldsymbol{o}_n$ in observation space. The feature vectors $\boldsymbol{o}_n$ capture the stylistic characteristics of the associated documents $\mathcal{D}_n$. Documents may consist of known words, as well as

Table 1: Average error rates and standard deviations for supervised authorship attribution experiments over the Amazon review data described in Section 3.2.2. A 5-fold cross-validation was used.

size of training set	tVAE	gVAE	SVM (linear)	SVM (RBF kernel)
20% (4,234 texts)	$\mathbf{5.86 \pm 0.60}$	6.24 ± 0.43	6.15 ± 0.37	6.41 ± 0.51
40% (8,468 texts)	$\mathbf{4.49 \pm 0.46}$	4.73 ± 0.40	5.05 ± 0.29	5.04 ± 0.66
60% (12,703 texts)	$\mathbf{3.90 \pm 0.34}$	4.25 ± 0.67	4.35 ± 0.24	4.41 ± 0.68
80% (16,937 texts)	$\mathbf{3.80 \pm 0.10}$	3.95 ± 0.43	3.92 ± 0.14	4.09 ± 0.32
100% (21,172 texts)	$\mathbf{3.53 \pm 0.24}$	3.86 ± 0.43	3.71 ± 0.38	3.70 ± 0.19

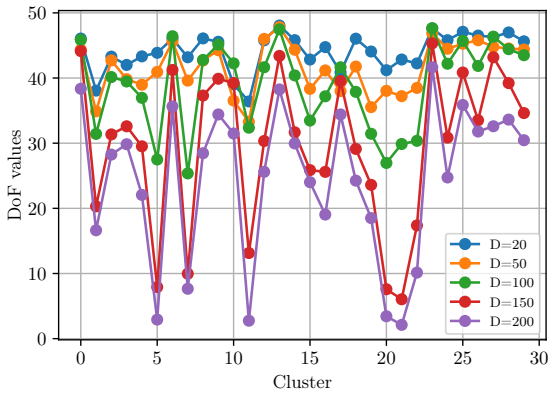


Figure 4: Degrees of freedom (DoF) ν_k for all clusters (i.e. authors) w.r.t. latent space dimension D.

unknown character concatenations embedded within multiple sentences. The core of ADHOMINEM is a two-level hierarchical attention-based bidirectional LSTM network (Hochreiter and Schmidhuber, 1997). Besides pre-trained word embeddings, ADHOMINEM also provides a characters-to-word encoding layer to take the specific uses of prefixes and suffixes as well as spelling errors into account.

3.2.2 Amazon Reviews Dataset

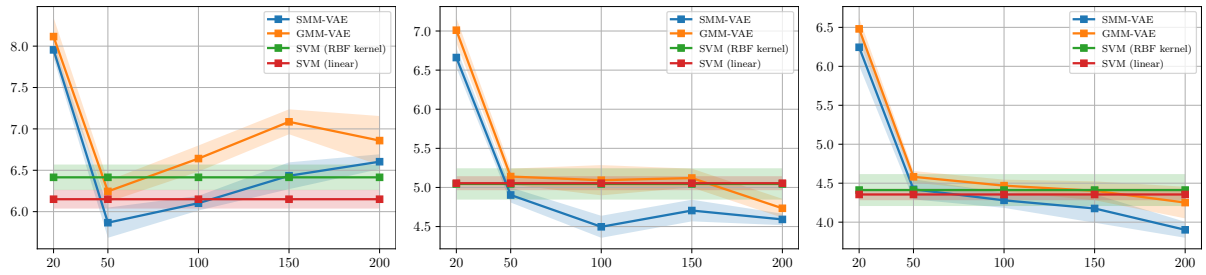
ADHOMINEM was trained on a large-scale corpus of short Amazon reviews. The dataset is described in (Boenninghoff et al., 2019b) and consists of $9,052,606$ reviews written by $784,649$ authors, with document lengths varying between 80 and 1000 tokens. For the evaluation of the proposed tVAE method, we randomly selected $21,172$ reviews written by 30 authors. All text from the selected authors had been excluded from the training procedure of ADHOMINEM. Each author contributed with at least 503 reviews and with a maximum of $1,000$ reviews.

3.2.3 Hyper-Parameter Tuning and Regularization

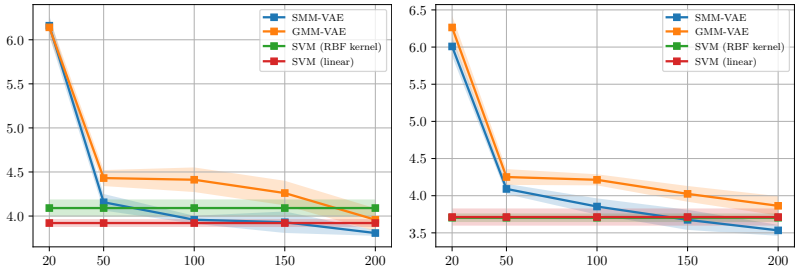
Encoder and decoder were fully connected feed-forward networks (with a `tanh` activation function) of the form L-H-D and D-H-L, respectively, where $L = 200$ defines the dimension of the observations, $H = \frac{D+L}{2}$ represents the number of hidden nodes and D is the latent space dimension. In all experiments, the Adam optimizer from (Kingma and Ba, 2014) was used to update the model parameters. Gradients were normalized so that their l_2-norm was less than or equal to 1. Furthermore, we added an l_1-regularization term, $J_{\boldsymbol{\phi},\boldsymbol{\theta}} = \beta \cdot \sum_{\forall(\boldsymbol{W},\boldsymbol{b})\in\{\boldsymbol{\phi},\boldsymbol{\theta}\}} \|\text{vec}(\boldsymbol{W})\|_1 + \|\boldsymbol{b}\|_1$, to reduce overfitting. The terms $\boldsymbol{W}, \boldsymbol{b}$ represent the weights and bias terms of the encoder/decoder networks. Our hyper-parameter tuning is based on a grid search over the following parameter-set combinations: stepsize $\alpha \in \{0.001, 0.002, 0.003, 0.004, 0.005\}$, $\sigma_k^2 \in \{0.001, 0.01, 0.1, 0.5, 0.9\}$, $D \in \{20, 50, 100, 150, 200\}$ and $\beta \in \{0.001, 0.005, 0.01, 0.05\}$.

3.2.4 Results

A 5-fold cross-validation was performed to evaluate the models in terms of average error rates. In a first step, we reserved 10% of the data as a development set and 10% of the data as a test set. In addition, we addressed the challenge of training the autoencoders with a smaller number of labeled data items by varying the size of the labeled training data from 20% to 100% (reviews were dropped out randomly). Using the best models found (depending on the hyper-parameters) we evaluated the performance of all methods with respect to the average error rate. Table 1 summarizes the results for our tVAE approach



(a) 20% of training data (b) 40% of training data (c) 60% of training data



(d) 80% of training data (e) 100% of training data

Figure 5: Average error rates as a function of the dimension D of the latent vectors $\boldsymbol{x}_n$ (from $D = 20$ to $D = 200 = L$) for various training data sizes.

versus the conventional gVAE method, as well as a linear and non-linear SVM. The lowest error rate for each setup is shown in bold face. It can be seen that our tVAE model is able to (slightly) outperform all baseline methods. For all methods, the performance gradually improves as the number of training documents is increased. In addition, Fig. 5 shows the performance results w.r.t. the dimension of the latent variable $\boldsymbol{x}_n$. It is apparent that the best choice for D increases when more reviews are added to the training set. For 20% of the training data, the optimal dimension is $D = 50$, for 40% we have $D = 100$ and for more than 60%, $D = 200$ yields the lowest error rate.

Fig. 4 displays the resulting, i.e. learned, degrees of freedom ν_k for all clusters (i.e. authors). The plot clearly shows that for smaller latent dimensions D, the cluster distributions are approximately Gaussian. With an increase in latent dimension, the mixture components become more heavy-tailed, making the Student-t distribution a better fit. Fig. 4, thus, provides a strong numerical justification for the move to the proposed tVAE model.

4 Conclusion

Variational autoencoders (VAEs) have proven their benefit in many tasks. They provide an attractive machine-learning framework that combines the strengths of neural-network training with the power of uncertainty metrics derived from statistical models. They can learn a low-dimensional manifold to summarize the most salient characteristics of data and provide a natural, statistical interpretation, both in the latent as well as in the observation space. In our work, we are addressing the question of whether VAEs can benefit from a statistical model that allows for more heavy-tailed distributions. The promise of heavy-tailed distributions is that susceptibility to outliers is curtailed and that the overall capacity of the model is improved.

Towards that goal, we have proposed and evaluated a VAE that is equipped with an embedded Student-t mixture model. It incorporates an assumption of Student-t distributed data into the joint learning mechanism for the latent manifold and its statistical distribution. Variational inference is performed by trying to simultaneously solve both tasks: jointly learning a nonlinear mapping to transform a given dataset of interest onto a (lower-dimensional) subspace *and* grouping the latent representations into meaningful categories.

We have derived a variational learning algorithm to accomplish this goal and we have shown its benefit for learning latent representations, both on synthetic data as well as a real-world authorship attribution task. Our more flexible model provided a capacity to obtain better results than an SVM-based classifier as well as a standard Gaussian VAE.

References

Najmeh Abiri and Mattias Ohlsson. 2019. Variational auto-encoders with Student's t-prior.

Cédric Archambeau and Michel Verleysen. 2007. Robust Bayesian clustering. *Neural Networks*, 20(1):129–138.

Christopher M. Bishop. 2006. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

B. Boenninghoff, R. M. Nickel, S. Zeiler, and D. Kolossa. 2019a. Similarity learning for authorship verification in social media. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2457–2461.

Benedikt Boenninghoff, Steffen Hessler, Dorothea Kolossa, and Robert M. Nickel. 2019b. Explainable authorship verification in social media via attention-based similarity learning. In *IEEE International Conference on Big Data, 2019, Los Angeles, CA, USA, December 9-12, 2019*.

Tim R. Davidson, Luca Falorsi, Nicola De Cao, Thomas Kipf, and Jakub M. Tomczak. 2018. Hyperspherical variational auto-encoders. *34th Conference on Uncertainty in Artificial Intelligence (UAI-18)*.

Nat Dilokthanakul, Pedro A. M. Mediano, Marta Garnelo, Matthew C. H. Lee, Hugh Salimbeni, Kai Arulkumaran, and Murray Shanahan. 2016. Deep unsupervised clustering with Gaussian mixture variational autoencoders. *CoRR*, abs/1611.02648.

Justin Domke and Daniel R Sheldon. 2018. Importance weighting and variational inference. In *Advances in Neural Information Processing Systems (NIPS)*, pages 4470–4479.

Janek Ebbers, Jahn Heymann, Lukas Drude, Thomas Glarner, Reinhold Häb-Umbach, and Bhiksha Raj. 2017. Hidden Markov model variational autoencoder for acoustic unit discovery. In *INTERSPEECH*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Zhuxi Jiang, Yin Zheng, Huachun Tan, Bangsheng Tang, and Hanning Zhou. 2017. Variational deep embedding: An unsupervised and generative approach to clustering. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI 2017, Melbourne, Australia, August 19-25, 2017*, pages 1965–1972.

Matthew J Johnson, David K Duvenaud, Alex Wiltschko, Ryan P Adams, and Sandeep R Datta. 2016. Composing graphical models with neural networks for structured representations and fast inference. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2946–2954. Curran Associates, Inc.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Diederik P. Kingma and Max Welling. 2013. Auto-encoding variational Bayes. *CoRR*, abs/1312.6114.

Wu Lin, Nicolas Hubacher, and Mohammad Emtiyaz Khan. 2018. Variational message passing with structured inference networks. In *International Conference on Learning Representations*.

Kevin P. Murphy. 2012. *Machine Learning: A Probabilistic Perspective*. The MIT Press.

Eric T. Nalisnick and Padhraic Smyth. 2017. Stick-breaking variational autoencoders. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Eric Nalisnick, Lars Hertel, and Padhraic Smyth. 2016. Approximate inference for deep latent gaussian mixtures. In *NIPS Workshop on Bayesian Deep Learning*, volume 2.

A. Rocha, W. J. Scheirer, C. W. Forstall, T. Cavalcante, A. Theophilo, B. Shen, A. R. B. Carvalho, and E. Stamatatos. 2017. Authorship attribution for social media forensics. *IEEE Trans. Inf. Forensic Secur.*, 12(1):5–33.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. 2015. Learning structured output representation using deep conditional generative models. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 3483–3491. Curran Associates, Inc.

Casper Kaae Sønderby, Tapani Raiko, Lars Maaløe, Søren Kaae Sønderby, and Ole Winther. 2016. Ladder variational autoencoders. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3738–3746.

Markus Svensén and Christopher M. Bishop. 2005. Robust Bayesian mixture modelling. *Neurocomput.*, 64:235–252.

Hiroshi Takahashi, Tomoharu Iwata, Yuki Yamanaka, Masanori Yamada, and Satoshi Yagi. 2018. Student-t variational autoencoder for robust density estimation. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*, pages 2696–2702. International Joint Conferences on Artificial Intelligence Organization, 7.

A. Theophilo, L. A. M. Pereira, and A. Rocha. 2019. A needle in a haystack? harnessing onomatopoeia and user-specific stylometrics for authorship attribution of micro-messages. In *Prc. ICASSP*, pages 2692–2696.

Michael Tschuggnall, Benjamin Murauer, and Günther Specht. 2019. Reduce & attribute: Two-step authorship attribution for large-scale problems. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 951–960. ACL.

Serena Yeung, Anitha Kannan, Yann Dauphin, and Li Fei-Fei. 2017. Tackling over-pruning in variational autoencoders. *CoRR*, abs/1706.03643.