KNIFE: KERNELIZED-NEURAL DIFFERENTIAL ENTROPY ESTIMATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Estimation of (differential) entropy and the related mutual information has been pursued with significant efforts by the machine learning community. To address shortcomings in previously proposed estimators for differential entropy, here we introduce KNIFE, a fully parameterized, differentiable kernel-based estimator of differential entropy. The flexibility of our approach also allows us to construct KNIFE-based estimators for conditional (on either discrete or continuous variables) differential entropy, as well as mutual information. We empirically validate our method on high-dimensional synthetic data and further apply it to guide the training of neural networks for real-world tasks. Our experiments on a large variety of tasks, including visual domain adaptation, textual fair classification, and textual fine-tuning demonstrate the effectiveness of KNIFE-based estimation.

1 INTRODUCTION

Learning tasks requires information (Principe et al., 2006) in the form of training data. Thus, information measures (Shannon, 1948) (*e.g.* entropy, conditional entropy and mutual information) have been a source of inspiration for the design of learning objectives in modern machine learning (ML) models (Linsker, 1989; Torkkola, 2006). Over the years, a plethora of estimators have been introduced to estimate the value of the aforementioned measures of information and they have been applied to many different problems, including information and coding theory, limiting distributions, model selection, design of experiment and optimal prior distribution, data disclosure, and relative importance of predictors (Ebrahimi et al., 2010). In these applications, traditional research focused on both developing new estimators and obtaining provable guarantees on the asymptotic behavior of these estimators (Liu et al., 2012; Verdú, 2019).

However, when used for training deep neural networks, additional *requirements* need to be satisfied. In particular, the estimator needs to be differentiable w.r.t. the data distribution **(R1)**, computationally tractable **(R2)**, and rapidly adapt to changes in the underlying distribution **(R3)**. For instance, Mutual Information (MI), a fundamental measure of dependence between variables, only became a popular (standalone or regularizing) learning objective for DNNs once estimators satisfying the above requirements were proposed (Poole et al., 2019; Barber & Agakov, 2003). Although MI is notoriously difficult to estimate in high dimensions (Kraskov et al., 2004; Pichler et al., 2020; McAllester & Stratos, 2020), these estimators have demonstrated promising empirical results in unsupervised representation learning (Krause et al., 2010; Bridle et al., 1992; Hjelm et al., 2019; Tschannen et al., 2020), discrete/invariant representations (Hu et al., 2017; Ji et al., 2019), generative modelling (Chen et al., 2016; Zhao et al., 2017), textual disentangling (Cheng et al., 2020b; Colombo et al., 2021), and applications of the Information Bottleneck (IB) method (Mahabadi et al., 2021; Devlin et al., 2018; Alemi et al., 2016) among others. Compared to MI, Differential Entropy (DE) has received less attention from the ML community while also having interesting applications.

In this paper, we focus on the problem of DE estimation as this quantity naturally appears in many applications (*e.g.* reinforcement learning (Shyam et al., 2019; Hazan et al., 2019; Ahmed et al., 2019; Kim et al., 2019), IB (Alemi et al., 2016), mode collapse (Belghazi et al., 2018)). Traditional estimators of DE often violate at least one of the requirements **(R1)** – **(R3)** listed above (*e.g.* k-nearest neighbor based estimators violate **(R1)**). As a consequence, the absence of DE estimator for arbitrary data distributions forces deep learning researchers to either restrict themselves to special cases where closed-form expressions for DE are available (Shyam et al., 2019) or use MI as a proxy

(Belghazi et al., 2018). In this work, we introduce a Kernelized Neural dIFferential Entropy (KNIFE) estimator, that satisfies the aforementioned requirements and addresses limitations of existing DE estimators (Schraudolph, 2004; McAllester & Stratos, 2020). Stemming from recent theoretical insights (McAllester & Stratos, 2020) that justify the use of DE estimators as building blocks to better estimate MI, we further apply KNIFE to MI estimation. In the context of deep neural networks with high dimensional data (*e.g.* image, text), KNIFE achieves competitive empirical results in applications where DE or MI is required.

1.1 CONTRIBUTIONS

Our work advances methods in DE and MI estimation in several ways.

1. We showcase limitation of the existing DE estimators proposed in Schraudolph (2004); McAllester & Stratos (2020) with respect to desirable properties required for training deep neural networks. To address these shortcomings, we introduce KNIFE, a fully learnable kernel-based estimator of DE. The flexibility of KNIFE allows us to construct KNIFE-based estimators for conditional DE, conditioning on either a discrete or continuous random variable.
2. We prove learnability under natural conditions on the underlying probability distribution. By requiring a fixed Lipschitz condition and bounded support we are not only able to provide an asymptotic result, but also a confidence bound in the case of a finite training set. This extends the consistency result by Ahmad & Lin (1976).
3. We validate on synthetic datasets (including multi-modal, non-Gaussian distributions), that KNIFE addresses the identified limitations and outperforms existing methods on both DE and MI estimation. In particular, KNIFE more rapidly adapts to changes in the underlying data distribution.
4. We conduct extensive experiments on natural datasets (including text and images) to compare KNIFE-based MI estimators to most recent MI estimators. First, we apply KNIFE in the IB principle to fine-tune a pretrained language model. Using KNIFE, we leverage a closed-form expression of a part of the training objective and achieve the best scores among competing MI estimators. Second, on fair textual classification, the KNIFE-based MI estimator achieves near perfect disentanglement (with respect to the private, discrete label) at virtually no degradation of accuracy in the main task. Lastly, in the challenging scenario of visual domain adaptation, where both variables are continuous, KNIFE-based MI estimation also achieves superior results.

1.2 EXISTENT METHODS AND RELATED WORKS

DE estimation. Existing methods for estimating DE fit into one of three categories (Beirlant et al., 1997; Hlaváčková-Schindler et al., 2007; Verdú, 2019): plug-in estimates (Ahmad & Lin, 1976; Györfi & Van der Meulen, 1987), estimates based on sample-spacings (Tarasenko, 1968), and estimates based on nearest neighbor distances (Kozachenko & Leonenko, 1987; Tsybakov & Van der Meulen, 1996); (Berrett et al., 2019). Our proposed estimator falls into the first category and we will thus focus here on previous work using that methodology. Excellent summaries of all the available methods can be found in the works (Beirlant et al., 1997; Hlaváčková-Schindler et al., 2007; Wang et al., 2009; Verdú, 2019). In Ahmad & Lin (1976), a first nonparametric estimator of DE was suggested and theoretically analyzed. It builds on the idea of kernel density estimation using Parzen-Rosenblatt windowing (Rosenblatt, 1956; Parzen, 1962). More detailed analysis followed (Joe, 1989; Hall & Morton, 1993) but the estimator remained essentially unchanged. Unfortunately, this classical literature is mostly concerned with appropriate regularity conditions that guarantee asymptotic properties of estimators, such as (asymptotic) unbiasedness and consistency. Machine learning applications, however, usually deal with a fixed—often very limited—number of samples.

Differentiable DE estimation. A first estimator that employed a differential learning rule was introduced in Viola et al. (1996). Indeed, the estimator proposed therein is optimized using stochastic optimization, it only used a single kernel with a low number of parameters. An extension that uses a *heteroscedastic* kernel density estimate, i.e., using different kernels at different positions, has been proposed in Schraudolph (2004). Still the number of parameters was quite low and varying means in the kernels or variable weights were not considered. Although the estimation of DE remained a topic of major interest as illustrated by recent works focusing on special classes of distributions (Kolchinsky & Tracey, 2017; Chaubey & Vu, 2021) and nonparametric estimators (Sricharan et al., 2013; Kandasamy et al., 2015; Moon et al., 2021), the estimator introduced in Schraudolph (2004) was not further refined and hardly explored in recent works.

Differentiable MI estimation. In contrast, there has been a recent surge on new methods for the estimation of the closely related MI between two random variables. The most prominent examples include unnormalized energy-based variational lower bounds (Poole et al., 2019), the lower bounds developed in Nguyen et al. (2010) using variational characterization of f-divergence, the MINE-estimator developed in Belghazi et al. (2018) from the Donsker-Varadhan representation of MI which can be also interpreted as an improvement of the plug-in estimator of Suzuki et al. (2008), the noise-contrastive based bound developed in van den Oord et al. (2018) and finally a contrastive upper bound (Cheng et al., 2020a). McAllester & Stratos (2020) point out shortcomings in other estimation strategies and introduce their own Differences of Entropies (DoE) method.

2 KNIFE

In this section we identify limitations of existing entropy estimators introduced in Schraudolph (2004); McAllester & Stratos (2020). Subsequently, we present KNIFE, which addresses these shortcomings.

2.1 LIMITATIONS OF EXISTING DIFFERENTIAL ENTROPY ESTIMATORS

Consider a continuous random variable $X \sim p$. Our goal is to estimate the DE $\mathrm{h}(X) := -\int p(x) \log p(x) \, \mathrm{d}x$. Given the intractability of this integral, we will rely on a Monte-Carlo estimate of $\mathrm{h}(X)$, using N i.i.d. samples $\mathcal{D}_{\mathrm{x}} = \{x_n\}_{n=1}^N$ to obtain

$$\widehat{\mathrm{h}}_{\text{ORACLE}}(\mathcal{D}_{\mathrm{x}}) := -\frac{1}{N} \sum_{n=1}^{N} \log p(x_n). \tag{1}$$

Unfortunately, assuming access to the true density p is often unrealistic, and we will thus construct an estimate $\hat{p}$ that can then be plugged into (1) instead of p. If $\hat{p}$ is smooth, the resulting plug-in estimator of DE is differentiable **(R1)**.

Assuming access to an additional—ideally independent—set of M i.i.d. samples $\mathcal{E} = \{x'_m\}_{m=1}^M$, we build upon the Parzen-Rosenblatt estimator (Rosenblatt, 1956; Parzen, 1962)

$$\hat{p}(x; w, \mathcal{E}) = \frac{1}{wM} \sum_{m=1}^{M} \kappa \left(\frac{x - x'_m}{w} \right), \tag{2}$$

where $w > 0$ denotes the bandwidth and κ is a kernel density. The resulting entropy estimator when replacing p in (1) by (2) was analyzed in Ahmad & Lin (1976). In Schraudolph (2004), this approach was extended using the kernel estimator

$$\hat{p}_{\text{SCHRAU.}}(x; \mathbf{A}, \mathcal{E}) := \frac{1}{M} \sum_{m=1}^{M} \kappa_{A_m}(x - x'_m), \tag{3}$$

where $\mathbf{A} := (A_1, \ldots, A_M)$ are (distinct, diagonal) covariance matrices and $\kappa_A(x) = \mathcal{N}(x; 0, A)$ is a centered Gaussian density with covariance matrix A.

The DoE method of McAllester & Stratos (2020) is a MI estimator that separately estimates a DE and a conditional DE. For DE, a simple Gaussian density estimate $\hat{p}_{\text{DoE}}(x; \boldsymbol{\theta}) = \kappa_A(x - \mu)$ is used, where $\boldsymbol{\theta} = (A, \mu)$ are the training parameters, the diagonal covariance matrix A and the mean μ.

While both SCHRAU. and DoE yield differentiable plug-in estimators for DE, they each have a major disadvantage. The strategy of Schraudolph (2004) fixes the kernel mean values at $\mathcal{E}$, which implies that the method cannot adapt to a shifting input distribution **(R3)**. On the other hand, DoE allows for rapid adaptation, but its simple structure makes it inadequate for the DE estimation of multi-modal densities. We illustrate these limitations in Section 3.1.

2.2 KNIFE ESTIMATOR

In KNIFE, the kernel density estimate is given by

$$\hat{p}_{\text{KNIFE}}(x; \boldsymbol{\theta}) := \sum_{m=1}^{M} u_m \kappa_{A_m}(x - a_m), \tag{4}$$

where $\boldsymbol{\theta} \coloneqq (\mathbf{A}, \mathbf{a}, \mathbf{u})$ and the additional parameters $0 \leq \mathbf{u} = (u_1, u_2, \ldots, u_M)$ with $\mathbf{1} \cdot \mathbf{u} = 1$ and $\mathbf{a} = (a_1, \ldots, a_M)$ are introduced. Note that $\hat{p}_{\text{KNIFE}}(x; \boldsymbol{\theta})$ is a smooth function of $\boldsymbol{\theta}$, and so is our proposed plug-in estimator

$$\widehat{\mathrm{h}}_{\text{KNIFE}}(\mathcal{D}_{\mathrm{x}}; \boldsymbol{\theta}) \coloneqq -\frac{1}{N} \sum_{n=1}^{N} \log \hat{p}_{\text{KNIFE}}(x_n; \boldsymbol{\theta}). \tag{5}$$

KNIFE combines the ideas of Schraudolph (2004); McAllester & Stratos (2020). It is differentiable and able to adapt to shifting input distributions, while capable of matching multi-modal distributions. Thus, as we will see in synthetic experiments, incorporating u_m and shifts a_m in the optimization enables the use of KNIFE in a non-stationary settings, where the distribution of X evolves over time.

Learning step: Stemming from the observation that, by the Law of Large Numbers (LLN),

$$\widehat{\mathrm{h}}_{\text{KNIFE}}(\mathcal{D}_{\mathrm{x}}, \boldsymbol{\theta}) \overset{\text{LLN}}{\approx} -\mathbb{E}\left[\log \hat{p}_{\text{KNIFE}}(X; \boldsymbol{\theta})\right] = \mathrm{h}(X) + \mathrm{D}_{\mathrm{KL}}(p \| \hat{p}_{\text{KNIFE}}(\,\cdot\,; \boldsymbol{\theta})) \geq \mathrm{h}(X), \tag{6}$$

we propose to learn the parameters $\boldsymbol{\theta}$ by minimizing $\widehat{\mathrm{h}}_{\text{KNIFE}}$, where $\mathcal{E}$ may be used to initialize $\mathbf{a}$. Although not strictly equivalent due to the Monte-Carlo approximation, minimizing $\widehat{\mathrm{h}}_{\text{KNIFE}}$ can be understood as minimizing the Kullback-Leibler (KL) divergence in (6), effectively minimizing the gap between $\widehat{\mathrm{h}}_{\text{KNIFE}}$ and $\mathrm{h}(X)$. In fact, $\widehat{\mathrm{h}}_{\text{KNIFE}}$ can also be interpreted as the standard maximum likelihood objective, widely used in modern machine learning. It is worth to mention that the KNIFE estimator is fully differentiable with respect to $\boldsymbol{\theta}$ and the optimization can be tackled by any gradient-based method (e.g., Adam (Kingma & Ba, 2014) or AdamW (Loshchilov & Hutter, 2017)).

2.3 Convergence Analysis

Note that the classical Parzen-Rosenblatt estimator $\widehat{\mathrm{h}}(\mathcal{D}_{\mathrm{x}}; w)$, where (2) is plugged into (1), is a special case of KNIFE. Thus, the convergence analysis provided in (Ahmad & Lin, 1976, Theorem 1) also applies and yields sufficient conditions for $\widehat{\mathrm{h}}_{\text{KNIFE}}(\mathcal{D}_{\mathrm{x}}, \boldsymbol{\theta}) \to \mathrm{h}(X)$. In Appendix C, we extend this result and, assuming that the underlying distribution p is compactly supported and L-Lipschitz continuous, the following theorem is proved.

Theorem 1. *For any $\delta > 0$, there exists a function $\varepsilon(N, M, w)$ such that, with probability at least $1 - \delta$, $\left|\widehat{\mathrm{h}}(\mathcal{D}_{\mathrm{x}}; w) - \mathrm{h}(X)\right| \leq \varepsilon(N, M, w)$. Additionally, $\varepsilon(N, M, w) \to 0$ as $M, N \to \infty$ and $w \to 0$ provided that*

$$Nw \to 0 \qquad \text{and} \qquad \frac{N^2 \log N}{w^2 M} \to 0, \tag{7}$$

where M and N denote the number of samples in $\mathcal{E}$ and $\mathcal{D}_{\mathrm{x}}$, respectively.

An explicit formula for $\varepsilon(N, M, w)$ is also given in Theorem 2 in Appendix C. For instance, Theorem 1 provides a bound on the speed of convergence for the consistency analysis in (Ahmad & Lin, 1976, Theorem 1).

2.4 Estimating Conditional Differential Entropy and Mutual Information

Similar to (McAllester & Stratos, 2020), the proposed DE estimator can be used to estimate other information measures. In particular, we can use KNIFE to construct estimators of conditional DE and MI. When estimating the conditional DE and MI for a pair of random variables $(X, Y) \sim p$, we not only use $\mathcal{D}_{\mathrm{x}} = \{x_n\}_{n=1}^N$, but also the according i.i.d. samples $\mathcal{D}_{\mathrm{y}} = \{y_n\}_{n=1}^N$, where (x_n, y_n) are drawn according to p.

Conditional Differential Entropy. We estimate conditional DE $\mathrm{h}(X|Y)$ by considering $\boldsymbol{\theta}$ to be a parameterized function $\boldsymbol{\Theta}(y)$ of y. Then all relations previously established naturally generalize and

$$\hat{p}_{\text{KNIFE}}(x|y; \boldsymbol{\Theta}) \coloneqq \hat{p}_{\text{KNIFE}}(x; \boldsymbol{\Theta}(y)), \quad \widehat{\mathrm{h}}_{\text{KNIFE}}(\mathcal{D}_{\mathrm{x}}|\mathcal{D}_{\mathrm{y}}; \boldsymbol{\Theta}) \coloneqq -\frac{1}{N} \sum_{n=1}^{N} \log \hat{p}_{\text{KNIFE}}(x_n|y_n; \boldsymbol{\Theta}). \tag{8}$$

Naturally, minimization of (6) is now performed over the parameters of $\boldsymbol{\Theta}$. If Y is a continuous random variable, we use an artificial neural network $\boldsymbol{\Theta}(y)$, taking y as its input. On the other hand, if $Y \in \mathcal{Y}$ is a discrete random variable, we have one parameter $\boldsymbol{\theta}$ for each $y \in \mathcal{Y}$, i.e., $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_y\}_{y \in \mathcal{Y}}$ and $\hat{p}_{\text{KNIFE}}(x|y; \boldsymbol{\Theta}) = \hat{p}_{\text{KNIFE}}(x; \boldsymbol{\Theta}(y)) = \hat{p}_{\text{KNIFE}}(x; \boldsymbol{\theta}_y)$.

Mutual Information. To estimate the MI between random variables X and Y (either discrete or continuous), recall that MI can be written as $\mathrm{I}(X;Y) = \mathrm{h}(X) - \mathrm{h}(X|Y)$. Therefore, we use the marginal and conditional DE estimators (5) and (8) to build a KNIFE-based MI estimator

$$\widehat{\mathrm{I}}_{\text{KNIFE}}(\mathcal{D}_{\mathrm{x}}, \mathcal{D}_{\mathrm{y}}; \boldsymbol{\theta}, \boldsymbol{\Theta}) := \widehat{\mathrm{h}}_{\text{KNIFE}}(\mathcal{D}_{\mathrm{x}}; \boldsymbol{\theta}) - \widehat{\mathrm{h}}_{\text{KNIFE}}(\mathcal{D}_{\mathrm{x}}|\mathcal{D}_{\mathrm{y}}; \boldsymbol{\Theta}). \tag{9}$$

3 EXPERIMENTS USING SYNTHETIC DATA

3.1 DIFFERENTIAL ENTROPY ESTIMATION

In this section we apply KNIFE for DE estimation, comparing it to (3), the method introduced in Schraudolph (2004), subsequently labeled "SCHRAU.". It is worth to mention that we did not perform the Expectation Maximization algorithm, as suggested in (Schraudolph, 2004), but instead opted to use the same optimization technique as for KNIFE to facilitate a fair comparison.

3.1.1 GAUSSIAN DISTRIBUTION

As a sanity check, we test KNIFE on multivariate normal data in moderately high dimensions, comparing it to SCHRAU. and DOE, which we trained with the exact same parameters. We performed these experiments with $d = 10$ and $d = 64$ dimensional data. KNIFE yielded the lowest bias and variance in both cases, despite DOE being perfectly adapted to matching a multivariate Gaussian distribution. Additional details can be found in Appendix A.1.

In order to use a DE estimation primitive in a machine learning system, it must be able to adapt to a changing input distribution during training **(R3)**. As already pointed out in Section 2.1, this is a severe limitation of SCHRAU., as re-drawing the kernel support $\mathcal{E}$ can be either impractical or at the very least requires a complete re-training of the entropy estimator. Whereas in (4), the kernel support $\mathbf{a}$ is trainable and it can thus adapt to a change of the input distribution. In order to showcase this ability, we utilize the approach of Cheng et al. (2020a) and successively decrease the entropy, observing how the estimator adapts. We perform this experiment with data of dimension $d = 64$ and repeatedly multiply the covariance matrix of the training vectors with a factor of $a = \frac{1}{2}$. The resulting entropy estimation is depicted in Figure 1. It is apparent that SCHRAU. suffers from a varying bias. The bias increases with decreasing variance, as the kernel support is fixed and cannot adapt as the variance of $\mathcal{D}_{\mathrm{x}}$ shrinks. DOE is perfectly adapted to a single Gaussian distribution and performs similar to KNIFE.

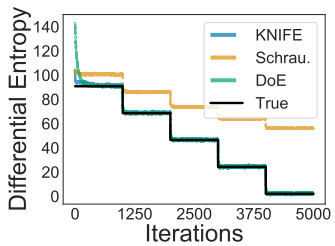


Figure 1: Estimating DE of Gaussian data with decreasing variance.

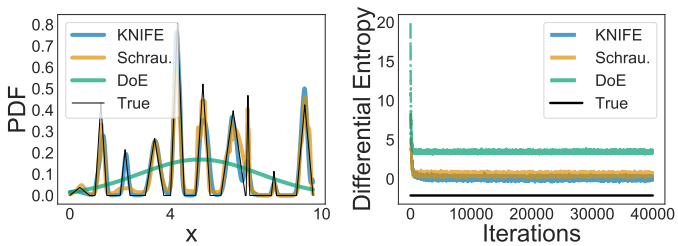


Figure 2: **Left**: PDF when estimating DE of a triangle mixture in 1 dimension. **Right**: Training run when estimating DE of a 2-component triangle mixture in 8 dimensions.

3.1.2 TRIANGLE MIXTURE

KNIFE is able to cope with distributions that have multiple modes. While (3) is also capable of matching multi-modal distributions, DOE is unable to do so, as it approximates any distribution with a multivariate Gaussian. We illustrate this by matching a mixture of randomly drawn triangle distributions. The resulting estimated PDFs as well as the ground truth when estimating the entropy of a 1-dimensional mixture of triangles with 10 components can be observed in Figure 2 (left). With increasing dimension the difficulty of this estimation rises quickly as in d dimensions, the resulting PDF of independent c-component triangle mixtures has c^d modes. To showcase the performance of KNIFE in this challenging task, we ran 10 training runs for DE estimation of 2-component triangle mixtures in 8 dimensions. An example training run is depicted in Figure 2 (right).

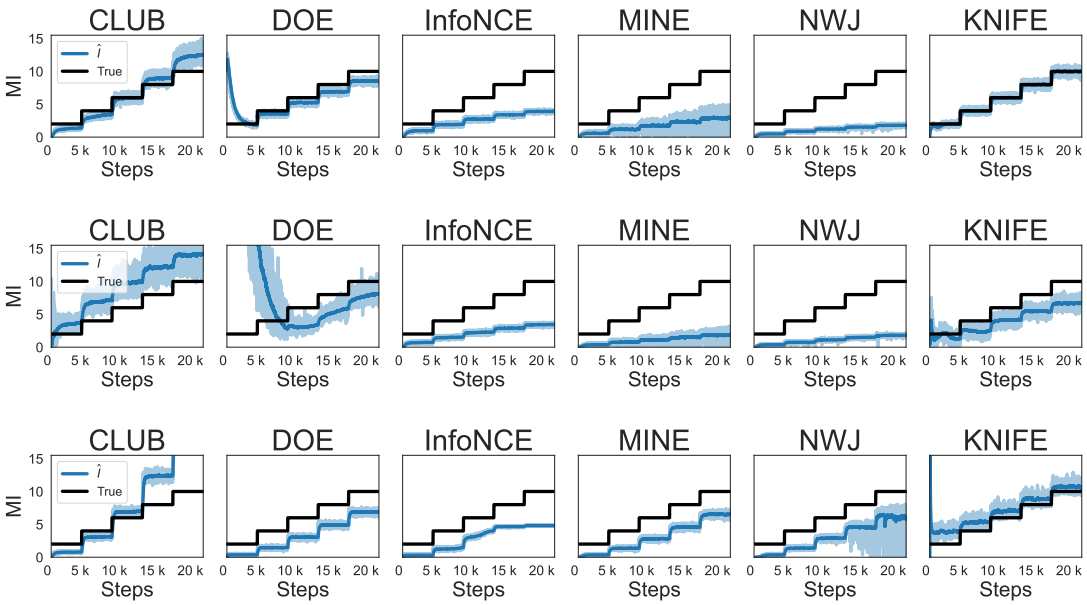


Figure 3: **Top**: Estimation of $\mathrm{I}(X^d; Y^d)$, where (X, Y) are multivariate Gaussian with correlation coefficient ρ_i in the i-th epoch and $d = 20$. **Middle**: Estimation of $\mathrm{I}(X^d; (Y^3)^d)$. **Bottom**: Estimation of $\mathrm{I}(X^d; Y^d)$ for uniform (X, E) and $Y = \rho_i X + \sqrt{1 - \rho_i^2} E$ in the i-th epoch.

3.2 Mutual Information Estimation

Multivariate Gauss We repeat the experiments in (Cheng et al., 2020a), stepping up the MI $\mathrm{I}(X^d; Y^d)$ between d i.i.d. copies of joint normal random variables (X, Y) by increasing their correlation coefficient, i.e., (X, Y) are multivariate Gaussian with correlation coefficient ρ_i in the i-th epoch. A training run is depicted in the top of Figure 3. As in (Cheng et al., 2020a), we also repeat the experiment, applying a cubic transformation to Y. The estimation of MI between d i.i.d. copies of X and Y^3 can be observed in the middle row of Figure 3. The MI is unaffected by this bijective transformation. In Appendix A.3, the bias and variance are depicted separately.

Sum of Uniformly Distributed Variables In order to test the ability of KNIFE to adapt to distributions substantially different from the Gaussian kernel shape, we apply it in MI estimation of $\mathrm{I}(X^d; Y^d)$ with uniformly distributed data. To this end, let X and E be centered, uniformly distributed random variables with $\mathbb{E}[X^2] = \mathbb{E}[E^2] = 1$ and define $Y = \rho_i X + \sqrt{1 - \rho_i^2} E$ in the i-th epoch. One training run with $d = 20$ is shown in Figure 3 (bottom). Details about the source distribution as well as details of the experiments can be found in Appendix A.3.

4 Experiments on Natural Data

In this section, we benchmark our proposed KNIFE-based MI estimator on three practical applications, spanning textual and visual data. We reproduce and compare our method to the most recent MI estimators including MINE (Belghazi et al., 2018), NWJ (Nguyen et al., 2010), InfoNCE (van den Oord et al., 2018), CLUB (Cheng et al., 2020a), and DoE (McAllester & Stratos, 2020). We do not explicitly include the SMILE estimator Song & Ermon (2019) in our comparison as it has the same gradient as NWJ.

Common notation: In all following applications, we will use $\Phi_\psi \colon \mathcal{X} \to \mathcal{Z}$ to denote an encoder, where $\mathcal{X}$ is the raw input space (i.e., texts or images), and $\mathcal{Z}$ denotes a lower dimensional continuous feature space. Additionally, we will use $C_\psi \colon \mathcal{Z} \to \mathcal{Y}$ to denote a shallow classifier from the latent space $\mathcal{Z}$ to a discrete or continuous target space $\mathcal{Y}$ for classification or regression, respectively. We will use ψ to denote the parameters of both models, Φ_ψ and C_ψ. CE denotes the cross entropy loss.

Table 1: Fine-tuning on GLUE. Following (Lee et al., 2019; Dodge et al., 2020), mean and variance are computed for 10 seeds.

	MRPC		STS-B		RTE
	F1	Accuracy	Pearson	Spearman	Accuracy
BERT (Devlin et al., 2018)	83.4 ± 0.9	88.2 ± 0.7	89.2 ± 0.4	88.8 ± 0.4	69.4 ± 0.4
CLUB (Cheng et al., 2020a)	85.0 ± 0.4	89.0 ± 0.7	89.7 ± 0.2	89.4 ± 0.1	70.7 ± 0.1
InfoNCE (van den Oord et al., 2018)	84.9 ± 0.8	88.9 ± 0.6	89.4 ± 0.4	89.7 ± 0.6	70.6 ± 0.1
MINE (Belghazi et al., 2018)	80.0 ± 2.5	85.0 ± 0.9	88.0 ± 0.7	88.0 ± 0.6	69.0 ± 0.9
NWJ (Nguyen et al., 2010)	84.6 ± 0.8	88.1 ± 0.7	89.8 ± 0.1	89.6 ± 0.2	69.6 ± 0.7
VUB/VIBERT (Alemi et al., 2016)	85.1 ± 0.5	89.1 ± 0.3	90.0 ± 0.2	89.5 ± 0.3	70.9 ± 0.1
DoE (McAllester & Stratos, 2020)	84.1 ± 0.2	88.3 ± 0.2	89.6 ± 0.2	89.5 ± 0.2	69.6 ± 0.2
KNIFE	**85.3** ± 0.1	**90.1** ± 0.1	**90.3** ± 0.0	**90.1** ± 0.0	**72.3** ± 0.2

4.1 INFORMATION BOTTLENECK FOR LANGUAGE MODEL FINETUNING

IB has recently been applied to fine-tune large-scale pretrained models (Mahabadi et al., 2021) such as BERT (Devlin et al., 2018) and aims at suppressing irrelevant features in order to reduce overfitting.

Problem statement. Given a textual input $X \in \mathcal{X}$ and a target label $Y \in \mathcal{Y}$, the goal is to learn the encoder $\Phi_{\boldsymbol{\psi}}$ and classifier $C_{\boldsymbol{\psi}}$, such that $\Phi_{\boldsymbol{\psi}}(X)$ retains little information about X, while still producing discriminative features, allowing the prediction of Y. Thus, the loss of interest is:

$$\mathcal{L} = \lambda \cdot \underbrace{\mathrm{I}(\Phi_{\boldsymbol{\psi}}(X); X)}_{\text{compression term}} - \underbrace{\mathrm{I}(\Phi_{\boldsymbol{\psi}}(X); Y)}_{\text{downstream term}}, \tag{10}$$

where λ controls the trade-off between the downstream and the compression terms.

Setup. Following Mahabadi et al. (2021) (relying on VUB), we work with the VIBERT model, which uses a Gaussian distribution as prior. $\Phi_{\boldsymbol{\psi}}$ is implemented as a stochastic encoder $\Phi_{\boldsymbol{\psi}}(X) = Z \sim \mathcal{N}(\mu_{\boldsymbol{\psi}}(X), \Sigma_{\boldsymbol{\psi}}(X))$. Details on the architecture of $\mu_{\boldsymbol{\psi}}$ and $\Sigma_{\boldsymbol{\psi}}$ can be found in Appendix B. The classifier $C_{\boldsymbol{\psi}}$ is composed of dense layers. To minimize $\mathcal{L}$, the second part of the objective (10) is bounded using the variational bound from Barber & Agakov (2003). Since we use a Gaussian prior, $\mathrm{h}(Z|X)$ can be expressed in closed form.[1] Thus, when using KNIFE, $\mathrm{I}(X;Z) = \mathrm{h}(Z) - \mathrm{h}(Z|X)$ can be estimated by using $\widehat{\mathrm{h}}_{\text{KNIFE}}$ to estimate $\mathrm{h}(Z)$. We compare this KNIFE-based MI estimator with aforementioned MI estimators and the variational upper bound (VUB). For completeness, we also compare against a BERT model trained by direct minimization of a CE loss.

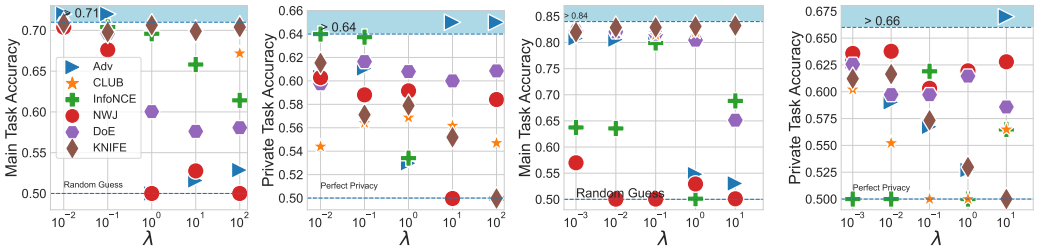
We closely follow the protocol of (Mahabadi et al., 2021) and work on the GLUE benchmark (Wang et al., 2018) originally composed of 5 datasets. However, following (Mahabadi et al., 2021), we choose to finetune neither on WNLI (Morgenstern & Ortiz, 2015) nor on CoLA (Warstadt et al., 2019) due to reported flaws in these datasets. The evaluation is carried out on the standard validation splits as the test splits are not available. Following standard practice (Liu et al., 2019; Yang et al., 2019), we report the accuracy and the F1 for MRPC, the accuracy for RTE and the Pearson and Spearman correlation coefficient for STS-B.

Results. Table 1 reports our results on the GLUE benchmark. We observe that KNIFE obtains the best results on all three datasets and the lowest variance on MRPC and STS-B. The use of a Gaussian prior in the stochastic encoder $\Phi_{\boldsymbol{\psi}}$ could explain the observed improvement of KNIFE-based estimation over MI-estimators such as CLUB, InfoNCE, MINE, DoE, or NWJ.

4.2 FAIR TEXTUAL CLASSIFICATION

In fair classification, we would like the model to take its decision without utilizing private information such as gender, age, or race. For this task, MI can be minimized to disentangle the output of the encoder Z and a private label $S \in \mathcal{S}$ (e.g., gender, age, or race).

[1] $\mathrm{h}(Z|X) = \frac{1}{2} \ln |\Sigma_{\boldsymbol{\psi}}(X)| + \frac{d}{2} \ln(2\pi e)$, where d is the dimension of X and $|\cdot|$ denotes the determinant.



(a) Y (sentiment) (b) S (sentiment) (c) Y (mention) (d) S (mention)

Figure 4: Results on the fair classification task for both main (Figures 4a and 4c) and private task (Figures 4b and 4d) for both mention and sentiment labels. Results of MINE are not reported because of instabilities that prevent the network from converging. Figures 4b and 4d are obtained by training an offline classifier to recover the protected attribute S from $\Phi_{\boldsymbol{\psi}}(X)$.

Problem Statement. Given an input text X, a discrete target label Y and a private label S, the loss is given by

$$\mathcal{L} = \underbrace{\mathrm{CE}(Y; \Phi_{\boldsymbol{\psi}}(X))}_{\text{downstream task}} + \lambda \cdot \underbrace{\mathrm{I}(\Phi_{\boldsymbol{\psi}}(X); S)}_{\text{disentangled}}, \tag{11}$$

where λ controls the trade-off between minimizing MI and CE loss. In this framework, a classifier is said to be fair or to achieve perfect privacy if no statistical information about S can be extracted from $\Phi_{\boldsymbol{\psi}}(X)$ by an adversarial classifier. Overall, a good model should achieve high accuracy on the main task (i.e., prediction of Y) while removing information about the protected attribute S. This information is measured by training an offline classifier to recover the protected attribute S from $\Phi_{\boldsymbol{\psi}}(X)$.

Setup. We compute the second term of (11) with competing MI estimators, as well as the model from Elazar & Goldberg (2018), which will be referred to as "Adv", as it utilizes an adversary to recover the private label from the latent representation Z. For KNIFE-based MI estimation, we use two DE estimators (as S is a binary label), following the approach outlined in Section 2.4. All derivations are detailed in Appendix B.
We follow the experimental setting from Elazar & Goldberg (2018); Barrett et al. (2019) and use two datasets from the DIAL corpus (Blodgett et al., 2016) (over 50 million tweets) where the protected attribute S is the race and the main labels are sentiment or mention labels. The mention label indicates whether a tweet is conversational or not. We follow the official split using 160 000 tweets for training and two additional sets composed of 10 000 tweets each for development and testing. In all cases, the labels S and Y are binary and balanced, thus a random guess corresponds to 50% accuracy.

Results. Figure 4 gathers results on the fair classification task. The upper dashed lines represent the (private and main) task accuracies when training a model with only the CE loss (case $\lambda = 0$ in (11)). This shows that the learned encoding $\Phi_{\boldsymbol{\psi}}(X)$ contains information about the protected attribute, when training is only performed for the main task. On both the sentiment and mention task, we observe that a KNIFE-based estimator can achieve perfect privacy (see Figures 4b and 4d) with nearly no accuracy loss in the main task (see Figures 4a and 4c). The other MI estimators exhibit different behavior. For sentiment labels, most MI estimators fail to reach perfect privacy (CLUB, NWJ, DOE, and Adv) while others (InfoNCE) achieve perfect privacy while degrading the main task accuracy (10% loss on main accuracy). For mention labels, CLUB can also reach perfect privacy with almost no degradation of the accuracy of the main task. Overall, it is worth noting that KNIFE-based MI estimation enables better control of the degree of disentanglement than the reported baselines.

4.3 UNSUPERVISED DOMAIN ADAPTATION

In unsupervised domain adaptation, the goal is to transfer knowledge from the source domain (S) with a potentially large number of labeled examples to a target domain (T), where only unlabeled examples are available.

Problem Statement. The learner is given access to labeled images from a source domain $(x_s, y) \sim (X_S, Y) \in \mathcal{X}_S \times \mathcal{Y}$ and unlabeled images from a target domain $x_t \sim X_T \in \mathcal{X}_T$. The goal is to

Table 2: Domain adaptation results: M (MNIST), MM (MNIST M), U (USPS), SV (SVHN), C (CIFAR10) and S (STL10). Results are averaged over 3 seeds.

	M → MM	S → C	U → M	M → U	C → S	SV → M	Mean
Source only	51.9 ±0.8	58.3 ±0.2	91.1 ±0.7	93.5 ±0.6	**72.3** ±0.5	54.7 ±2.8	70.3 ±0.9
CLUB	79.1 ±2.2	59.9 ±1.9	96.0 ±0.2	96.8 ±0.5	71.6 ±1.3	83.8 ±3.4	81.2 ±1.7
DoE	**82.2** ±2.6	58.9 ±0.8	97.2 ±0.3	94.2 ±0.9	68.8 ±1.4	86.4 ±5.4	81.3 ±1.9
InfoNCE	77.3 ±0.5	61.0 ±0.1	97.4 ±0.2	97.0 ±0.3	70.6 ±0.8	89.2 ±4.1	82.1 ±1.0
MINE	76.7 ±0.4	61.2 ±0.3	**97.7** ±0.1	97.3 ±0.1	70.8 ±1.0	91.8 ±0.8	82.6 ±0.4
NWJ	77.1 ±0.6	61.2 ±0.3	97.6 ±0.1	97.3 ±0.5	72.1 ±0.7	91.4 ±0.8	82.8 ±0.5
Knife	78.7 ±0.7	**61.8** ±0.5	**97.7** ±0.3	**97.4** ±0.4	71.2 ±1.8	**93.2** ±0.2	**83.4** ±0.6

learn a classification model $\{\Phi_{\boldsymbol{\psi}}, C_{\boldsymbol{\psi}}\}$ that generalizes well to the target domain. Training models on the supervised source data only results in domain-specific latent representations $\Phi_{\boldsymbol{\psi}}(X)$ leading to poor generalization (when X is chosen randomly from $\{X_S, X_T\}$). In order to make the latent representations as domain-agnostic as possible, we follow the information-theoretic method proposed by Gholami et al. (2020), and used in Cheng et al. (2020a). The idea is to learn an additional binary model $\{\Phi^d_{\boldsymbol{\nu}}, C^d_{\boldsymbol{\nu}}\}$, whose goal it is to guess the domain $D \in \{0, 1\}$ of X. The latent representation learned by $\Phi^d_{\boldsymbol{\nu}}$ will therefore contain all the domain-specific information that we would like the main encoder $\Phi_{\boldsymbol{\psi}}$ to discard. In other words, we would like $\Phi_{\boldsymbol{\psi}}(X)$ and $\Phi^d_{\boldsymbol{\nu}}(X)$ to be completely disentangled, which naturally corresponds to the minimization of $\mathrm{I}(\Phi_{\boldsymbol{\psi}}(X); \Phi^d_{\boldsymbol{\nu}}(X))$. Concretely, the domain classifier is trained to minimize the CE between domain labels D and its own predictions, whereas the main classifier is trained to properly classify support samples while minimizing the MI between $\Phi_{\boldsymbol{\psi}}(X)$ and $\Phi^d_{\boldsymbol{\nu}}(X)$. Using $f^d_{\boldsymbol{\nu}} \coloneqq C^d_{\boldsymbol{\nu}} \circ \Phi^d_{\boldsymbol{\nu}}$ and $f_{\boldsymbol{\psi}} \coloneqq C_{\boldsymbol{\psi}} \circ \Phi_{\boldsymbol{\psi}}$, the objectives are

$$\min_{\boldsymbol{\nu}} \ \mathrm{CE}(D; f^d_{\boldsymbol{\nu}}(X)) \qquad \text{and} \qquad \min_{\boldsymbol{\psi}} \ \mathrm{CE}(Y; f_{\boldsymbol{\psi}}(X_S)) + \lambda \cdot \mathrm{I}(\Phi_{\boldsymbol{\psi}}(X); \Phi^d_{\boldsymbol{\nu}}(X)). \tag{12}$$

Setup. The different MI estimators are compared based on their ability to guide training by estimating $\mathrm{I}(\Phi_{\boldsymbol{\psi}}(X); \Phi^d_{\boldsymbol{\nu}}(X))$ in (12). We follow the setup of Cheng et al. (2020a) as closely as possible, and consider a total of 6 source/target scenarios formed with MNIST (LeCun & Cortes, 2010), MNIST-M (Ganin et al., 2016), SVHN (Netzer et al., 2011), CIFAR-10 (Krizhevsky et al., 2009), and STL-10 (Coates et al., 2011) datasets. We reproduce all methods and allocate the same budget for hyper-parameter tuning to every method. The exhaustive list of hyper-parameters can be found in Appendix B.

Results. Results are presented in Table 2. The Knife-based estimator is able to outperform MI estimators in this challenging scenario where both $\Phi_{\boldsymbol{\psi}}(X)$ and $\Phi^d_{\boldsymbol{\nu}}(X)$ are continuous.

5 Concluding Remarks

We introduced Knife, a fully learnable, differentiable kernel-based estimator of differential entropy, designed for deep learning applications. We constructed a mutual information estimator based on Knife and showcased several applications. Knife is a general purpose estimator and does not require any special properties of the learning problem. It can thus be incorporated as part of any training objective, where differential entropy or mutual information estimation is desired. In the case of mutual information, one random variable may even be discrete.

Despite the fundamental challenges in the problem of differential entropy estimation, beyond limitations arising from the use of a finite number of samples, Knife has demonstrated promising empirical results in various representation learning tasks.

Future work will focus on improving the confidence bounds given in Theorem 1. In particular, tailoring them towards Knife using tools from (Birge & Massart, 1995; Singh & Poczos, 2014). Another potential extension is direct estimation of the gradient of entropy, when $\hat{p}_{\text{Knife}}(x; \boldsymbol{\theta})$ has been learned (Mohamed et al., 2020; Song et al., 2020). This could be applied after the learning phase of Knife and is left for future work.

REFERENCES

I. Ahmad and Pi-Erh Lin. A nonparametric estimation of the entropy for absolutely continuous distributions. *IEEE Trans. Inf. Theory*, 22(3):372–375, May 1976.

Zafarali Ahmed, Nicolas Le Roux, Mohammad Norouzi, and Dale Schuurmans. Understanding the impact of entropy on policy optimization. 97:151–160, 09–15 Jun 2019.

Alexander A. Alemi, Ian Fischer, Joshua V. Dillon, and Kevin Murphy. Deep variational information bottleneck. *arXiv*, abs/1612.00410, 2016.

David Barber and Felix Agakov. The im algorithm: A variational approach to information maximization. In *Proceedings of the 16th International Conference on Neural Information Processing Systems*, NIPS'03, pp. 201–208, Cambridge, MA, USA, 2003. MIT Press.

Maria Barrett, Yova Kementchedjhieva, Yanai Elazar, Desmond Elliott, and Anders Søgaard. Adversarial removal of demographic attributes revisited. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pp. 6331–6336, 2019.

Jan Beirlant, Edward J. Dudewicz, László Györfi, and Edward C. Van der Meulen. Nonparametric entropy estimation: An overview. *International Journal of Mathematical and Statistical Sciences*, 6(1):17–39, 1997.

Ishmael Belghazi, Sai Rajeswar, Aristide Baratin, R. Devon Hjelm, and Aaron C. Courville. MINE: mutual information neural estimation. *arXiv*, abs/1801.04062, 2018.

Thomas B Berrett, Richard J Samworth, and Ming Yuan. Efficient multivariate entropy estimation via k-nearest neighbour distances. *The Annals of Statistics*, 47(1):288–318, 2019.

Lucien Birge and Pascal Massart. Estimation of Integral Functionals of a Density. *The Annals of Statistics*, 23(1):11 – 29, 1995. doi: 10.1214/aos/1176324452. URL https://doi.org/10.1214/aos/1176324452.

Su Lin Blodgett, Lisa Green, and Brendan O'Connor. Demographic dialectal variation in social media: A case study of african-american english. *arXiv*, abs/1608.08868, 2016.

John Bridle, Anthony Heading, and David MacKay. Unsupervised classifiers, mutual information and 'phantom targets. In J. Moody, S. Hanson, and R. P. Lippmann (eds.), *Advances in Neural Information Processing Systems*, volume 4. Morgan-Kaufmann, 1992.

Yogendra P Chaubey and Nhat Linh Vu. On the estimation of entropy for non-negative data. *Journal of Statistical Theory and Practice*, 15(2), 2021.

Xi Chen, Yan Duan, Rein Houthooft, John Schulman, Ilya Sutskever, and Pieter Abbeel. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. pp. 2180–2188, 2016.

Pengyu Cheng, Weituo Hao, Shuyang Dai, Jiachang Liu, Zhe Gan, and Lawrence Carin. CLUB: A contrastive log-ratio upper bound of mutual information. 119, 13–18 Jul 2020a.

Pengyu Cheng, Martin Renqiang Min, Dinghan Shen, Christopher Malon, Yizhe Zhang, Yitong Li, and Lawrence Carin. Improving disentangled text representation learning with information-theoretic guidance. *arXiv*, abs/2006.00693, 2020b.

Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2011.

Pierre Colombo, Chloe Clavel, and Pablo Piantanida. A novel estimator of mutual information for learning to disentangle textual representations. *arXiv*, abs/2105.02685, 2021.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv*, abs/1810.04805, 2018.

Jesse Dodge, Gabriel Ilharco, Roy Schwartz, Ali Farhadi, Hannaneh Hajishirzi, and Noah Smith. Fine-tuning pretrained language models: Weight initializations, data orders, and early stopping. *arXiv*, abs/2002.06305, 2020.

Nader Ebrahimi, Ehsan S. Soofi, and Refik Soyer. Information measures in perspective. *International Statistical Review*, 78(3):383–412, 2010.

Yanai Elazar and Yoav Goldberg. Adversarial removal of demographic attributes from text data. *arXiv*, abs/1808.06640, 2018.

Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The journal of machine learning research*, 17(1), 2016.

Behnam Gholami, Pritish Sahu, Ognjen Rudovic, Konstantinos Bousmalis, and Vladimir Pavlovic. Unsupervised multi-target domain adaptation: An information theoretic approach. *IEEE Transactions on Image Processing*, 29, 2020.

László Györfi and Edward C. Van der Meulen. Density-free convergence properties of various estimators of entropy. *Computational Statistics & Data Analysis*, 5(4):425–436, 1987.

Peter Hall and Sally Morton. On the estimation of entropy. *Annals of the Institute of Statistical Mathematics*, 1993.

Elad Hazan, Sham Kakade, Karan Singh, and Abby Van Soest. Provably efficient maximum entropy exploration. 97:2681–2691, 09–15 Jun 2019.

R Devon Hjelm, Alex Fedorov, Samuel Lavoie-Marchildon, Karan Grewal, Phil Bachman, Adam Trischler, and Yoshua Bengio. Learning deep representations by mutual information estimation and maximization. 2019.

Katerina Hlaváčková-Schindler, Milan Paluš, Martin Vejmelka, and Joydeep Bhattacharya. Causality detection based on information-theoretic approaches in time series analysis. *Physics Reports*, 441 (1), 2007.

Weihua Hu, Takeru Miyato, Seiya Tokui, Eiichi Matsumoto, and Masashi Sugiyama. Learning discrete representations via information maximizing self-augmented training. In Doina Precup and Yee Whye Teh (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1558–1567. PMLR, 06–11 Aug 2017.

Xu Ji, Joao F. Henriques, and Andrea Vedaldi. Invariant information clustering for unsupervised image classification and segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

Harry Joe. Estimation of entropy and other functionals of a multivariate density. *Annals of the Institute of Statistical Mathematics*, 41(4):683–697, 1989.

Kirthevasan Kandasamy, Akshay Krishnamurthy, Barnabas Poczos, Larry Wasserman, and james m robins. Nonparametric von mises estimators for entropies, divergences and mutual informations. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015. URL https://proceedings.neurips.cc/paper/2015/file/06138bc5af6023646ede0e1f7c1eac75-Paper.pdf.

Hyoungseok Kim, Jaekyeom Kim, Yeonwoo Jeong, Sergey Levine, and Hyun Oh Song. EMI: exploration with mutual information. 97:3360–3369, 2019.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. 2014.

Artemy Kolchinsky and Brendan D Tracey. Estimating mixture entropy with pairwise distances. *Entropy*, 19(7), 2017.

LF Kozachenko and Nikolai N Leonenko. Sample estimate of the entropy of a random vector. *Problemy Peredachi Informatsii*, 23(2):9–16, 1987.

Alexander Kraskov, Harald Stögbauer, and Peter Grassberger. Estimating mutual information. *Phys. Rev. E*, 69, Jun 2004.

Andreas Krause, Pietro Perona, and Ryan Gomes. Discriminative clustering by regularized information maximization. In J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta (eds.), *Advances in Neural Information Processing Systems*, volume 23. Curran Associates, Inc., 2010.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images. 2009.

Yann LeCun and Corinna Cortes. MNIST handwritten digit database. 2010.

Cheolhyoung Lee, Kyunghyun Cho, and Wanmo Kang. Mixout: Effective regularization to finetune large-scale pretrained language models. *arXiv*, abs/1909.11299, 2019.

Ralph Linsker. How to generate ordered maps by maximizing the mutual information between input and output signals. In *Neural Comput.*, 1989.

Han Liu, Larry Wasserman, and John Lafferty. Exponential concentration for mutual information estimation with application to forests. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv*, abs/1907.11692, 2019.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv*, abs/1711.05101, 2017.

Rabeeh Karimi Mahabadi, Yonatan Belinkov, and James Henderson. Variational information bottleneck for effective low-resource fine-tuning. *ICLR. OpenReview. net*, 2021.

David McAllester and Karl Stratos. Formal limitations on the measurement of mutual information. In Silvia Chiappa and Roberto Calandra (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 875–884. PMLR, 26–28 Aug 2020.

Shakir Mohamed, Mihaela Rosca, Michael Figurnov, and Andriy Mnih. Monte carlo gradient estimation in machine learning. *J. Mach. Learn. Res.*, 21(132):1–62, 2020.

Kevin R. Moon, Kumar Sricharan, and Alfred O. Hero. Ensemble estimation of generalized mutual information with applications to genomics. *IEEE Transactions on Information Theory*, 67(9): 5963–5996, 2021. doi: 10.1109/TIT.2021.3100108.

Leora Morgenstern and Charles Ortiz. The winograd schema challenge: Evaluating progress in commonsense reasoning. In *Twenty-Seventh IAAI Conference*, 2015.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

XuanLong Nguyen, Martin J Wainwright, and Michael I Jordan. Estimating divergence functionals and the likelihood ratio by convex risk minimization. *IEEE Transactions on Information Theory*, 2010.

Liam Paninski. Estimation of entropy and mutual information. *Neural computation*, 15(6), 2003.

Emanuel Parzen. On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076, 1962.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019.

Georg Pichler, Pablo Piantanida, and Günther Koliander. On the estimation of information measures of continuous distributions. *arXiv*, abs/2002.02851, 2020.

Ben Poole, Sherjil Ozair, Aaron Van Den Oord, Alex Alemi, and George Tucker. On variational bounds of mutual information. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5171–5180. PMLR, 09–15 Jun 2019.

Jose C Principe, Dongxin Xu, John Fisher, and Simon Haykin. *Information theoretic learning*. Citeseer, 2006.

Shauli Ravfogel, Yanai Elazar, Hila Gonen, Michael Twiton, and Yoav Goldberg. Null it out: Guarding protected attributes by iterative nullspace projection. *arXiv preprint arXiv:2004.07667*, 2020.

Murray Rosenblatt. Remarks on some nonparametric estimates of a density function. *Ann. Math. Statist.*, 27(3):832–837, 1956.

N. N. Schraudolph. Gradient-based manipulation of nonparametric entropy estimates. *IEEE Transactions on Neural Networks*, 15(4), 2004.

C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3): 379–423, July 1948.

Pranav Shyam, Wojciech Jaśkowski, and Faustino Gomez. Model-based active exploration. In Kamalika Chaudhuri and Ruslan Salakhutdinov (eds.), *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5779–5788. PMLR, 09–15 Jun 2019.

Shashank Singh and Barnabas Poczos. Exponential concentration of a density functional estimator. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL https://proceedings.neurips.cc/paper/2014/file/af5afd7f7c807171981d443ad4f4f648-Paper.pdf.

Jiaming Song and Stefano Ermon. Understanding the limitations of variational mutual information estimators. *arXiv*, abs/1910.06222, 2019.

Yang Song, Sahaj Garg, Jiaxin Shi, and Stefano Ermon. Sliced score matching: A scalable approach to density and score estimation. In *Uncertainty in Artificial Intelligence*, pp. 574–584. PMLR, 2020.

Kumar Sricharan, Dennis Wei, and Alfred O. Hero. Ensemble estimators for multivariate entropy estimation. *IEEE Transactions on Information Theory*, 59(7):4374–4388, 2013. doi: 10.1109/TIT.2013.2251456.

Taiji Suzuki, Masashi Sugiyama, Jun Sese, and Takafumi Kanamori. Approximating mutual information by maximum likelihood density ratio estimation. In *New challenges for feature selection in data mining and knowledge discovery*, pp. 5–20. PMLR, 2008.

F.P. Tarasenko. On the evaluation of an unknown probability density function, the direct estimation of the entropy from independent observations of a continuous random variable, and the distribution-free entropy test of goodness-of-fit. *Proceedings of the IEEE*, 56(11):2052–2053, 1968.

Kari Torkkola. *Information-Theoretic Methods in "Feature Extraction: Foundations and Applications"*, pp. 167–185. Springer, Berlin, Heidelberg, 2006.

Michael Tschannen, Josip Djolonga, Paul K. Rubenstein, Sylvain Gelly, and Mario Lucic. On mutual information maximization for representation learning. In *International Conference on Learning Representations*, 2020.

A.B. Tsybakov. *Introduction to Nonparametric Estimation*. Springer Series in Statistics. Springer New York, 2008.

Alexandre B. Tsybakov and E. C. Van der Meulen. Root-n consistent estimators of entropy for densities with unbounded support. *Scandinavian Journal of Statistics*, pp. 75–83, 1996.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *arXiv*, abs/1807.03748, 2018.

Sergio Verdú. Empirical estimation of information measures: A literature guide. *Entropy*, 21(8), 2019.

Paul Viola, Nicol N Schraudolph, and Terrence J Sejnowski. Empirical entropy manipulation for real-world problems. *Advances in neural information processing systems*, 1996.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv*, abs/1804.07461, 2018.

Qing Wang, Sanjeev R. Kulkarni, and Sergio Verdú. Universal estimation of information measures for analog sources. *Foundations and Trends in Communications and Information Theory*, 5(3): 265–353, 2009.

Alex Warstadt, Amanpreet Singh, and Samuel R Bowman. Neural network acceptability judgments. *Transactions of the Association for Computational Linguistics*, 7:625–641, 2019.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Huggingface's transformers: State-of-the-art natural language processing. *arXiv*, abs/1910.03771, 2019.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. Xlnet: Generalized autoregressive pretraining for language understanding. *arXiv*, abs/1906.08237, 2019.

Shengjia Zhao, Jiaming Song, and Stefano Ermon. InfoVAE: Information maximizing variational autoencoders. *arXiv*, abs/1706.02262, 2017.

APPENDIX

A EXPERIMENTAL DETAILS OF EXPERIMENTS WITH SYNTHETIC DATA

Implementation of KNIFE in PyTorch (Paszke et al., 2019) is rather straightforward. The constraint on the weights $\mathbf{u}$ can be satisfied by applying a `softmax` transformation. The covariance matrices were parameterized by the lower-triangular factor in the Cholesky decomposition of the precision matrices, guaranteeing the definiteness constraint to be satisfied.

A.1 DIFFERENTIAL ENTROPY ESTIMATION OF GAUSSIAN DATA

In Section 3.1.1, the estimation of the entropy $\mathrm{h}(X) = \frac{d}{2}\log 2\pi e$ for $X \sim \mathcal{N}(\mathbf{0}, I_d)$ was performed with the hyperparameters given in Table 3. The mean error and its empirical standard deviation are reported in Table 5 over 20 runs, where an independently drawn evaluation set with the same size as the training set is used. At $d = 10$ we have the entropy $h = \frac{d}{2}\log 2\pi e = 14.19$, while for the higher dimension, $d = 64$ we find $h = 90.81$.

In the experiment depicted in Figure 1, entropy is decreased after every epoch by letting $X_i \sim \mathcal{N}(\mathbf{0}, a^i I_d)$, where $i = 0, \ldots, 4$ is the epoch index. That is, $X_i = \sqrt{a^i} G^d$, where G is a standard normal random variable, resulting in an decrease of the DE by $\Delta = -\frac{d}{2}\log a \approx 22.18$ for $a = \frac{1}{2}$ with every epoch. We start at $\mathrm{h}(X_0) = \frac{d}{2}\log 2\pi e \approx 90.81$ and successively decrease until $\mathrm{h}(X_4) = \mathrm{h}(X_0) + 4\Delta \approx 2.1$. Additional parameters can be found in Table 4.

Computational Resources. Training was performed on an NVidia V100 GPU. Taken together, training for the first experiments of entropy estimation in dimensions $d = 10, 64$, as well as the experiment depicted in Figure 1 used GPU time of less than 5 minutes.

A.2 DIFFERENTIAL ENTROPY ESTIMATION OF TRIANGLE MIXTURES

In Section 3.1.2, we perform an estimation of the entropy of c-component triangle mixture distributions. The PDF of such a c-component triangle-mixture, is given by

$$p(x) = \sum_{i=1}^{c} w_i \Lambda_{s_i}\left(x - i - \frac{1}{2}\right), \tag{13}$$

where $\Lambda_s(x) := \frac{1}{s}\max\{0, 2 - 4s|x|\}$ is a centered triangle PDF with width $s > 0$. The scales $\mathbf{s} = (s_1, \ldots, s_c)$ and weights $\mathbf{w} = (w_1, \ldots, w_c)$ satisfy $0 < s_i, w_i < 1$ and $\sum_{i=1}^{c} w_i = 1$. Before the experiment, we choose $\mathbf{w}$ uniformly at random from the c-probability simplex and the scales are chosen uniformly at random in $[0.1, 1.0]$. An example for $c = 10$ is the true PDF depicted in Figure 2

Table 3: Experimental details of first experiment in Section 3.1.1.

Parameter	Value
Source Distribution X	$X \sim \mathcal{N}(\mathbf{0}, I_d)$
Dimension d	10 and 64
Optimizer	Adam
Learning Rate	0.01
Batch Size N	128
Kernel Size M	128
Iterations per epoch	200
Epochs	1
Runs	20

Table 4: Experimental details of the experiment depicted in Figure 1.

Parameter	Value
Source Distribution X	$X \sim \mathcal{N}(\mathbf{0}, a^i I_d)$ for $i = 0, \ldots, 4$
Dimension d	64
Factor a	$\frac{1}{2}$
Optimizer	Adam
Learning Rate	0.01
Batch Size N	128
Kernel Size M	128
Iterations per epoch	1000
Epochs	5
Runs	1

Table 5: Results of first experiment in Section 3.1.1 with Gaussian data. We provide the average distance $|h - \widehat{h}|$ and the empirical standard deviation. Experimental details are given in Table 3.

$\lvert h - \widehat{h}\rvert$	$d = 10$	$d = 64$
DoE	0.8388 ± 1.0045	3.3170 ± 1.8281
Schrau.	0.7301 ± 0.0428	9.8919 ± 0.1604
Knife	$\mathbf{0.0461 \pm 0.0139}$	$\mathbf{2.8045 \pm 0.0796}$

Table 6: Experimental details of the experiment resulting in the PDF in Figure 2 (left).

Parameter	Value
Source Distribution X	c-component triangle mixtures
Components c	10
Dimension d	1
Optimizer	Adam
Learning Rate	0.1
Batch Size N	128
Kernel Size M	128
Iterations per epoch	100
Epochs	10
Runs	1

Table 7: Experimental details of the experiment resulting in the training depicted in Figure 2 (right).

Parameter	Value
Source Distribution X	c-component triangle mixtures
Components c	2
Dimension d	8
Optimizer	Adam
Learning Rate	0.001
Batch Size N	128
Kernel Size M	128
Iterations per epoch	1000
Epochs	20
Runs	10

(left). For $d > 1$, we perform the estimation on d i.i.d. copies. Note that the triangle mixture with c components in d-dimensional space has c^d modes, i.e., the support can be partitioned into c^d disjoint components.

The parameters of the experiment yielding Figure 2 (left) are given in Table 6, while the details of the experiment depicted in Figure 2 (right) can be found in Table 7. In the latter experiment, over ten runs, entropy was estimated to an accuracy of 1.6563 ± 0.8528 by Knife, accurate to 2.4445 ± 0.5439 using (3) and with an accuracy of 7.1070 ± 2.7984 by DoE. This is the mean absolute error and its empirical standard deviation over all 10 runs, where the evaluation set was drawn independently from the training set and has the same size as the training set.

Computational Resources. Training was performed on an NVidia V100 GPU. Training in $d = 1$ dimension, that resulted in Figure 2 (left) can be performed in seconds, while all training required for producing Figure 2 (right) used approximately 1.5 hours of GPU time.

A.3 Mutual Information Estimation

In Section 3.2, we estimate $\mathrm{I}(X^d; Y^d)$ and $\mathrm{I}(X^d; (Y^3)^d)$ where (X, Y) are multivariate correlated Gaussian distributions with correlation coefficient ρ_i in the i-th epoch. Subsequently, we estimate $\mathrm{I}(X^d; Y^d)$ where $X, E \sim \mathcal{U}[-\sqrt{3}, \sqrt{3}]$ are independent and Y is given by $Y = \rho_i X + \sqrt{1 - \rho_i^2} E$. In both cases, ρ_i is chosen such that $\mathrm{I}(X^d; Y^d) = 2i$ in the i-th epoch.

All neural networks are randomly initialized. The bias, variance, and MSE during training as a function of the MI, can be observed in Figure 5.

The estimation is performed in 10 runs, randomly choosing the training meta-parameters as proposed by McAllester & Stratos (2020). In Figure 3 (bottom), we present the best run for each method, selected by distance from the true MI at the end of training. The bias, variance, and MSE during training, as a function of the MI, can be observed in Figure 6. Details about the source distribution as well as details of the experiments can be found in Table 8. During experimentation it turned out to be

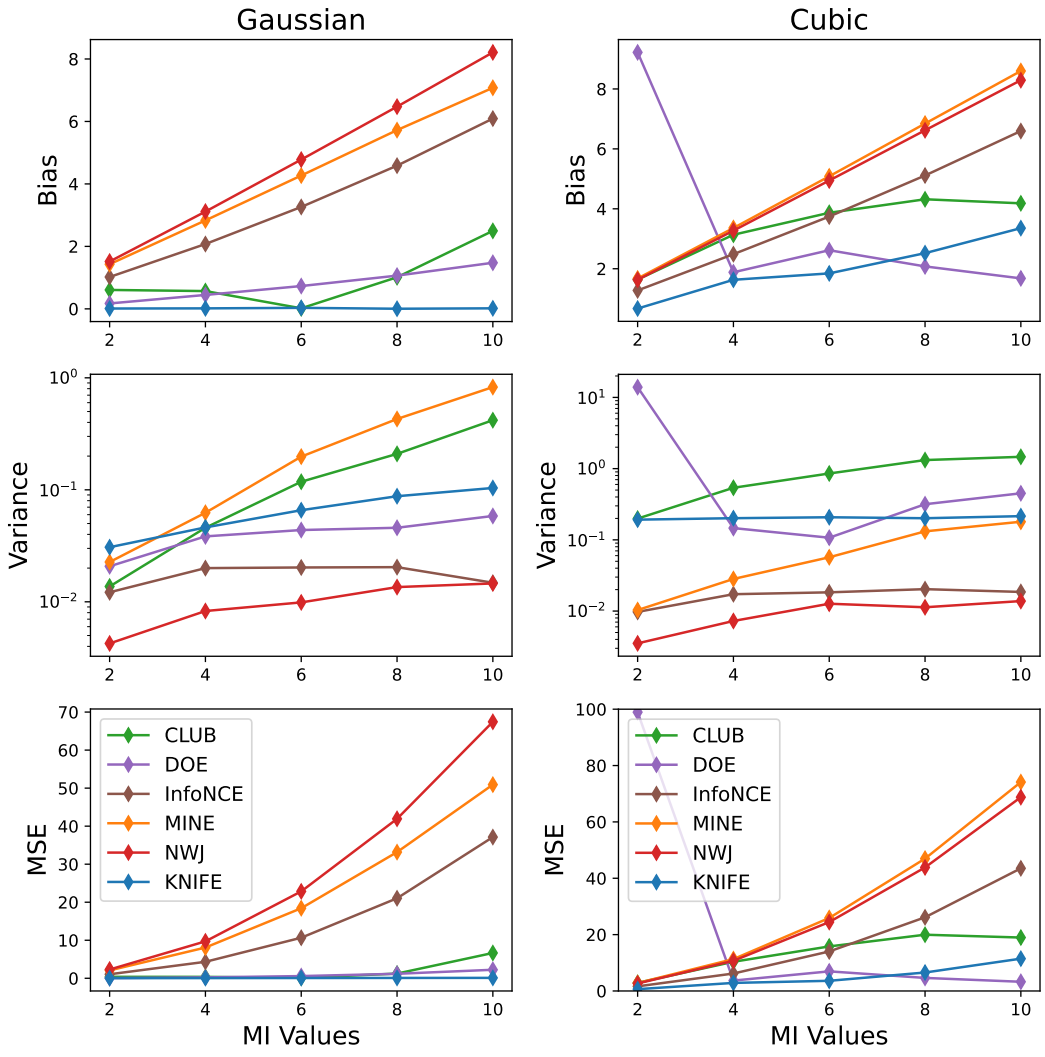


Figure 5: **Left**: Estimation of $\mathrm{I}(X;Y)$; **Right**: Estimation of $\mathrm{I}(X;Y^3)$ (cubic transformation).

Table 8: Experimental details of the training depicted in Figure 3 (bottom).

Parameter	Value
Dimension d	20
Optimizer	Adam
Learning Rates	0.01, 0.003, 0.001, 0.0003
Batch Size N	128
Kernel Size M	128
Iterations per epoch	20 000
Epochs	1
Runs	10

beneficial to train the parameters $\boldsymbol{\Theta}$ and $\boldsymbol{\theta}$ in (9) separately and substantially increase the learning rate for the training of $\boldsymbol{\theta}$. Thus, we increase the learning rate for the training of $\boldsymbol{\theta}$ by a factor of 10^3.

Model Architecture for $\boldsymbol{\Theta}$. We utilize the feed-forward architecture, also used in McAllester & Stratos (2020). It is a simple architecture with two linear layers, one hidden layer using $\tanh$ activation, immediately followed by an output layer. The number of neurons in the hidden layer is a meta-parameter selected randomly from $\{64, 128, 256\}$ for each training run. Three models with this architecture are used for the three parameters $(\mathbf{A}, \mathbf{a}, \mathbf{u})$, as described by (4), where only the output dimension is changed to fit the parameter dimension.

Computational Resources. Training was performed, using about 6 hours of GPU time on an NVidia V100 GPU to carry out the experiment depicted in Figure 3 (bottom).

B Experimental Details of Experiments on Natural Data

B.1 On the parameter update

In Section 4, we rely on two different types of models: pretrained (e.g., fine tuning with VIBERT) and randomly initialized (e.g., in fair classification and domain adaptation). When working with randomly initialized networks the parameters are updated. However, it is worth noting that in the litterature the pre-trained model parameters (*i.e.* ψ) are not always updated (see Ravfogel et al. (2020)). In our experiments: (i) We always update the parameters (even for pretrained models), and (ii) we did not change the way the parameters were updated in concurrent works (to ensure fair comparison). Specifically,

- for language model finetuning (Appendix B.2), we followed Mahabadi et al. (2021) and did a joint update;
- for the fair classification task (Appendix B.3), we followed common practice and used the algorithm described in Algorithm 1 which rely on an alternated update;
- for the domain adaptation task (Appendix B.4), we followed common practice and used a joint method.

B.2 Information Bottleneck for Language Model Finetuning

For this experiment we follow the experimental setting introduced in Mahabadi et al. (2021) and work with the GLUE data[2].

Model Architecture. We report in Table 9, the multilayer perceptron (MLP) used to compute the compressed sentence representations produced by BERT. Variance and Mean MLP networks are composed of fully connected layers.

[2]see https://gluebenchmark.com/faq

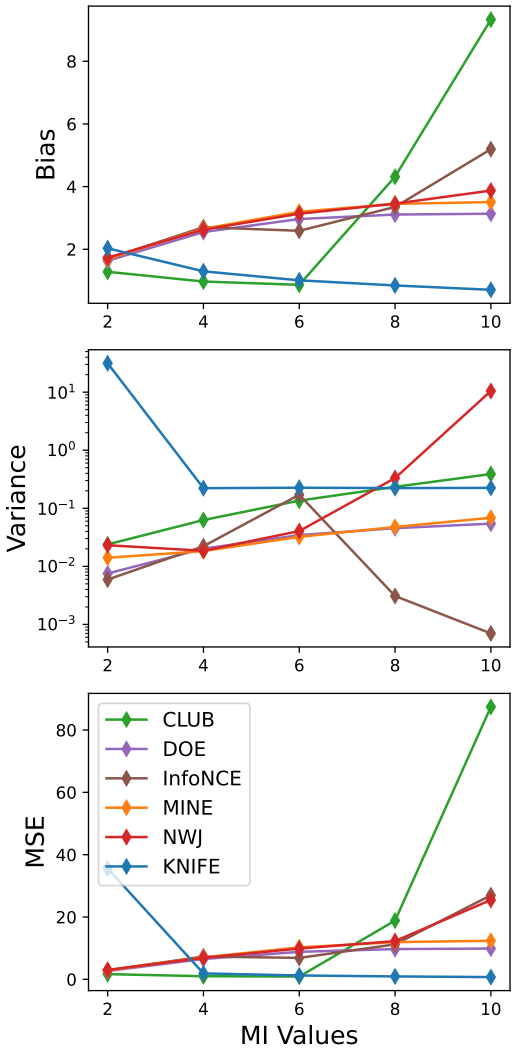


Figure 6: Bias, variance, and MSE for MI estimation on uniformly distributed data.

Algorithm 1 Disentanglement using a MI-based regularizer

INPUT Labelled training set $\mathcal{D} = \{(x_j, s_j, y_j) \forall j \in [n+1, N]\}$; independent set of samples $\mathcal{E}$; θ parameters KNIFE; ψ parameters of network.
INITIALIZE parameters θ, ψ
OPTIMIZATION
while (θ, ψ) not converged **do**
    for $i \in [1, \text{Unroll}]$ **do** ▷ Learning Step for KNIFE
        Sample a batch $\mathcal{B}$ from $\mathcal{E}$
        Update θ using ((9)).
    end for
    Sample a batch $\mathcal{B}'$ from $\mathcal{D}$
    Update θ with $\mathcal{B}'$ ((11)).
end while
OUTPUT Encoder and classifier weights ψ

Table 9: Architecture of the model used in the IB finetuning experiment. We use ReLU as an activation function.

Layer type	Input shape	Output shape
Fully connected	768	$\frac{2304+K}{4}$
Fully connected	$\frac{2304+K}{4}$	$\frac{768+K}{2}$

Table 10: Experimental details on Information Bottleneck.

Parameter	Value
Learning Rate	See Appendix B.2
Optimizer	AdamW
Warmup Steps	0.0
Dropout	0.0
Batch Size	32

Table 11: Datasets from the GLUE as used in our experiments.

	#Labels	Train	Val.	Test
RTE	2	2.5k	0.08k	3k
STS-B	1 (regression)	5.8k	1.5k	1.4k
MRPC	2	3.7k	0.4k	1.7k

Model Training. For model training, all models are trained for 6 epochs and we use early stopping (best model is selected on validation set error). For IB, λ is selected in $\{10^{-4}, 10^{-5}, 10^{-6}\}$ and K is selected in $\{144, 192, 288, 384\}$. We follow (Alemi et al., 2016) where the posterior is averaged over 5 samples and a linear annealing schedule is used for λ. Additional hyper-parameters are reported in Table 10.

Dataset Statistics. Table 11 reports the statistics of the dataset used in our finetuning experiment.

Computational Resources. For all these experiments we rely on NVidia-P100 with 16GB of RAM. To complete the full grid-search on 10 seeds and on the three datasets, approximately 1.5k hours are required.

B.3 Fair Textual Classification

In this section, we gather the experimental details for the textual fair classification task.

B.3.1 Details of the Knife-based Estimator

In this experiment, we estimate the MI between a continuous random variable, namely $Z = \Phi_\psi(X)$, and a discrete variable, denoted by $S \in \mathcal{S} = \{1, 2, \ldots, |\mathcal{S}|\}$. We follow the strategy outlined in Section 2.4 for estimating the conditional DE $\mathrm{h}(Z|S)$. However, we will reuse the estimate of the conditional PDF $\hat{p}(z|s; \boldsymbol{\Theta})$ to compute an estimate of the DE as

$$\mathrm{h}(Z) \approx -\frac{1}{N} \sum_{n=1}^{N} \log \left(\sum_{s \in \mathcal{S}} \hat{p}_{\text{KNIFE}}(z_n|s; \boldsymbol{\Theta}) \hat{p}(s) \right), \tag{14}$$

where $\hat{p}(s) = \frac{1}{N}|\{n : s_n = s\}|$ is used to indicate the empirical distribution of S in the training set $\mathcal{D}_{\text{s}}$.[3] In our experiments, with $|\mathcal{S}| = 2$, we found that estimating the DE $\mathrm{h}(Z)$ based on the KNIFE estimator learnt for $\mathrm{h}(Z|S)$ increases the stability of training. We adopted the same strategy for DoE.

B.3.2 Experimental Details

Model Architecture. For the encoder, we use a bidirectionnal GRU with two layers with hidden and input dimension set to 128. We use LeakyReLU as the activation function. The classification head is composed of fully connected layers of input dimension 256. We use a learning rate of 0.0001 for AdamW. The dropout rate is set to 0.2. The number of warmup steps is set to 1000.

[3] As we work with balanced batches, we will have $\hat{p}(s) = \frac{1}{|\mathcal{S}|}$.

Computational Resources. For all these experiments, we rely on NVIDIA-P100 with 16GB of RAM. Each model is trained for 30k steps. The model with the lowest MI is selected. The training of a single network takes around 3 hours.

B.4 Unsupervised Domain Adaptation

We follow the experimental setup given in Cheng et al. (2020a) as closely as possible, i.e., we pick hyperparameters given in the paper, or if not provided, those set in the code:[4]

Model Training. We use Adam optimizer for all modules with a learning rate of 0.001. Batch size is set to 128. We set the weighting parameter $\lambda = 0.1$. The original code of Cheng et al. (2020a) uses 15 000 training iterations, but we found most methods had not properly converged at this stage, and hence use 25 000 iterations instead. Similar to other experiments, we set the kernel size $M = 128$.

Model Architecture. Table 12 summarizes the architectures used for the different modules. For the MI network of each method, the best configuration, based on the validation set of the first task MNIST $\rightarrow$ MNIST-M, is chosen among 4 configurations: with or without LayerNorm and with ReLU or $\tanh$ activation.

Computational Resources. For these experiments, we used a cluster of NVIDIA-V100 with 16GB of RAM. Each training (i.e., 25k iterations) on a single task requires on average 2 hours. Given that we have 6 tasks, and repeat the training for 3 different seeds, on average 36 hours computation time is required for each method.

C Bounding the Error

In the following, fix $L > 0$ and let $\mathcal{P}_L$ be the set of L-Lipschitz PDFs supported[5] on $\mathcal{X} := [0,1]^d$, i.e., $\int_{\mathcal{X}} p(\mathbf{x})\,\mathrm{d}\mathbf{x} = 1$, and

$$\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^d : |p(\mathbf{x}) - p(\mathbf{y})| \leq L\|\mathbf{x} - \mathbf{y}\| \tag{15}$$

for $p \in \mathcal{P}_L$, where[6] $\|\mathbf{x}\| := \sum_k |x_k|$.

Assume $p \in \mathcal{P}_L$ and let κ be a PDF supported on $\mathcal{X}$. In order to show that estimation of $\mathrm{h}(\mathbf{X})$ is achievable, we use a standard Parzen-Rosenblatt estimator $\hat{p}(\mathbf{x}; w) := \frac{1}{Nw}\sum_{m=1}^{M} \kappa\left(\frac{\mathbf{x}-\mathbf{X}'_m}{w}\right)$, as in (2). The entropy estimate is then defined by the empirical average

$$\widehat{\mathrm{h}}(\mathcal{D}_{\mathrm{x}}; w) := -\frac{1}{N}\sum_{n=1}^{N} \log \hat{p}(\mathbf{X}_n; w). \tag{16}$$

Further, define the following quantities, which are assumed to be finite:

$$p_{\max} := \max\{p(\mathbf{x}) : \mathbf{x} \in \mathcal{X}\}, \tag{17}$$

$$C_1 := \int p(\mathbf{x}) \log^2 p(\mathbf{x}) \mathrm{d}\mathbf{x}, \tag{18}$$

$$C_2 := L \int \|\mathbf{u}\|\, \kappa(\mathbf{u}) \mathrm{d}\mathbf{u}, \tag{19}$$

$$K_{\max} := \max\{\kappa(\mathbf{x}) : \mathbf{x} \in \mathcal{X}\}. \tag{20}$$

Note that it is easily seen that $p_{\max} \leq \frac{L}{2}$ and $C_1 \leq \max\left\{p_{\max}\log^2 p_{\max}, 4e^{-2}\right\}$ by our assumptions. The requirement $C_2, K_{\max} < \infty$ represents a mild condition on the kernel function κ.

We can now show the following.

[4] https://github.com/Linear95/CLUB/tree/master/MI_DA.

[5] Any known compact support suffices. An affine transformation then yields $\mathcal{X} = [0,1]^d$, while possibly resulting in a different Lipschitz constant.

[6] The ℓ_1 norm is only chosen to facilitate subsequent computations. By the equivalence of norms on $\mathbb{R}^d$, any norm suffices.

Table 12: Architectures used for the Unsupervised Domain Adaptation experiments. For the MI network of each method, we chose the best performing configuration between with or without LayerNorm layer and best activation between ReLU and $\tanh$, using the validation set of MNIST-M.

Encoder (both Φ and Φ^d)

Layer type	Input shape	Output shape	Details
Convolution sequence	(3, H, W)	(64, H, W)	Cf. below
Noisy downsampling	(64, H, W)	(64, H // 2, W // 2)	Cf. below
Convolution sequence	(64, H // 2, W // 2)	(64, H // 2, W // 2)	Cf. below
Noisy downsampling	(64, H // 2, W // 2)	(64, H // 4, W // 4)	Cf. below
Convolution sequence	(64, H // 4, W // 4)	(64, H // 4, W // 4)	Cf. below
Global Average Pool	(64, H // 4, W // 4)	(64,)	-

Main classifier C

Layer type	Input shape	Output shape
Fully connected	(64,)	(10,)

Domain classifier C^d

Layer type	Input shape	Output shape
Fully connected	(64,)	(2,)

Convolution sequence

Layer type	Input shape	Output shape	Parameters
2D convolution	(3, H, W)	(64, H, W)	3x3, 64 channels, Stride=1, Padding=1
2D BatchNorm	(3, H, W)	(64, H, W)	-
Activation	(3, H, W)	(64, H, W)	LeakyRelu 0.1
2D convolution	(64, H, W)	(64, H, W)	3x3, 64 channels, Stride=1, Padding=1
2D BatchNorm	(64, H, W)	(64, H, W)	-
Activation	(64, H, W)	(64, H, W)	LeakyRelu 0.1
2D convolution	(64, H, W)	(64, H, W)	3x3, 64 channels, Stride=1, Padding=1
2D BatchNorm	(64, H, W)	(64, H, W)	-
Activation	(64, H, W)	(64, H, W)	LeakyRelu 0.1

Noisy downsampling

Layer type	Input shape	Output shape	Parameters
MaxPool	(64, H, W)	(64, H // 2, H // 2)	2x2, Stride=2
Dropout	(64, H // 2, W // 2)	(64, H // 2, H // 2)	p=0.5
Noise	(64, H // 2, W // 2)	(64, H // 2, H // 2)	Gaussian with $\sigma = 1$

MI network

Layer type	Input shape	Output shape	Details
LayerNorm	$(C_{in},)$	$(C_{in},)$	Optional
Fully connected	$(C_{in},)$	(64,)	Activation = [ReLU, $\tanh$]
LayerNorm	$(C_{in},)$	(64,)	Optional
Fully connected	(64,)	$(C_{out},)$	Optional

Theorem 2. *With probability greater than $1-\delta$ we have*

$$|\,\mathrm{h}(X) - \widehat{\mathrm{h}}(\mathcal{D}_{\mathrm{x}}; w)| \leq -\log\left(1 - \frac{3NK_{\max}}{w\delta}\sqrt{\frac{\log\frac{6N}{\delta}}{2M}} - \frac{3NC_2 w}{\delta}\right) + \sqrt{\frac{3C_1}{N\delta}}, \tag{21}$$

if the expression in the logarithm is positive.

In particular, the estimation error approaches zero as $N \to \infty$ if $w = https://www.overleaf.com/project/609d4ea5edff1a4d656d35fbw(N) \to 0$, $M = M(N) \to \infty$

are chosen such that

$$Nw \to 0, \tag{22}$$

$$\frac{N^2 \log N}{w^2 M} \to 0. \tag{23}$$

We prove Theorem 2 in several Lemmas.

Lemma 3. *Fix $\delta > 0$ and $\mathbf{x}_0 \in \mathcal{X}$. Then, with probability greater than $1 - \delta$,*

$$|p(\mathbf{x}_0) - \hat{p}(\mathbf{x}_0)| \leq \frac{K_{\max}}{w} \sqrt{\frac{\log \frac{2}{\delta}}{2M}} + C_2 w. \tag{24}$$

Proof. First, we can show that

$$|\mathbb{E}[\hat{p}(\mathbf{x}_0)] - p(\mathbf{x}_0)| \leq C_2 w \tag{25}$$

as κ is a kernel of order $\ell = 0$ and $p \in \mathcal{P}(1, L)$, using the notation of (Tsybakov, 2008, Sec. 1.2.1) and the ℓ_1 norm. We thus obtain (25) by (Tsybakov, 2008, Prop. 1.2). Next, note that

$$|\mathbb{E}[\hat{p}(\mathbf{x}_0)] - \hat{p}(\mathbf{x}_0)| \leq \frac{K_{\max}}{w} \sqrt{\frac{\log \frac{2}{\delta}}{2M}} \tag{26}$$

holds as the requirements of McDiarmid's inequality (Paninski, 2003, Sec. 3) are satisfied with $c_j = \frac{K_{\max}}{Mw}$ and thus $\mathrm{P}\{|\mathbb{E}[\hat{p}(\mathbf{x}_0)] - \hat{p}(\mathbf{x}_0)| \geq \varepsilon\} \leq \delta$ with

$$\varepsilon = \frac{K_{\max}}{w} \sqrt{\frac{\log \frac{2}{\delta}}{2M}}. \tag{27}$$

Combining (25) and (26) gives (24). □

Lemma 4. *For any continuous random variable $\mathbf{X}$ supported on $\mathcal{X}$ and $a \geq 0$, we have*

$$\mathrm{P}\{p(\mathbf{X}) \leq a\} \leq a. \tag{28}$$

Proof. We apply Markov's inequality to the random variable $Y = \frac{1}{p(X)}$ and observe that

$$\mathrm{P}\{p(\mathbf{X}) \leq a\} = \mathrm{P}\{Y \geq a^{-1}\} \leq \mathrm{vol}(\mathcal{X}) a = a. \tag{29}$$

□

Lemma 5. *If $x > 0$, $y \geq a > 0$, $0 < a < 1$, and $|x - y| \leq \delta < a$, then*

$$|\log x - \log y| \leq \log \frac{a}{a - \delta} = -\log\left(1 - \frac{\delta}{a}\right). \tag{30}$$

Proof. **Case $x \geq y$.** We can write $y = a + b$ and $x = y + c = a + b + c$ for $b \geq 0$ and $0 \leq c \leq \delta < a$.

$$\left|\log \frac{x}{y}\right| = \log\left(1 + \frac{c}{a+b}\right) \tag{31}$$

$$\leq \log\left(1 + \frac{c}{a}\right) \leq \log\left(1 + \frac{\delta}{a}\right). \tag{32}$$

Furthermore,

$$\log\left(\frac{a}{a-\delta}\right) - \log\left(1 + \frac{\delta}{a}\right) = \log \frac{1}{(a+\delta)(a-\delta)} \tag{33}$$

$$= \log \frac{1}{a^2 - \delta^2} \tag{34}$$

$$\geq \log \frac{1}{a^2} = -2 \log a > 0. \tag{35}$$

Case $x < y$**.** Here, we can write $y = a + b$ and $x = y - c = a + b - c$ for $b \geq 0$ and $0 \leq c \leq \delta < a$.

$$\left|\log \frac{x}{y}\right| = \log \frac{y}{x} \tag{36}$$

$$= \log \left(\frac{a+b}{a+b-c}\right) \tag{37}$$

$$\leq \log \left(\frac{a}{a-c}\right) \tag{38}$$

$$\leq \log \left(\frac{a}{a-\delta}\right) = -\log \left(1 - \frac{\delta}{a}\right). \tag{39}$$

□

Proof of Theorem 2. We apply Lemma 3 N times and use the union bound to show that with probability greater than $1 - \frac{\delta}{3}$ we have for every $n \in [N]$

$$|p(\mathbf{X}_n) - \hat{p}(\mathbf{X}_n)| \leq \frac{K_{\max}}{w}\sqrt{\frac{\log \frac{6N}{\delta}}{2M}} + C_2 w. \tag{40}$$

Similarly, by Lemma 4, we have with probability greater than $1 - \frac{\delta}{3}$ that

$$p(\mathbf{X}_n) \geq \frac{\delta}{3N} \tag{41}$$

for all $n \in [N]$.

Again by the union bound, we have that with probability greater than $1 - \frac{2\delta}{3}$ both (40) and (41) hold, and in turn, by Lemma 5, we have

$$\left|\widehat{\mathrm{h}}(\mathcal{D}_{\mathrm{x}}; w) + \frac{1}{N}\sum_{n=1}^{N} \log p(\mathbf{X}_n)\right| = \left|\frac{1}{N}\sum_{n=1}^{N} \log \frac{p(\mathbf{X}_n)}{\hat{p}(\mathbf{X}_n)}\right| \tag{42}$$

$$\leq -\log \left(1 - \frac{\frac{K_{\max}}{w}\sqrt{\frac{\log \frac{6N}{\delta}}{2M}} + C_2 w}{\frac{\delta}{3N}}\right) \tag{43}$$

$$= -\log \left(1 - \frac{3NK_{\max}}{w\delta}\sqrt{\frac{\log \frac{6N}{\delta}}{2M}} - \frac{3NC_2 w}{\delta}\right), \tag{44}$$

provided the argument in the logarithm is positive. Finally, we have the upper bound on the variance

$$\mathbb{E}\left[\left(\mathrm{h}(\mathbf{X}) + \frac{1}{N}\sum_{n=1}^{N} \log p(\mathbf{X}_n)\right)^2\right] = \frac{1}{N^2}\sum_{n=1}^{N} \mathbb{E}[(\mathrm{h}(\mathbf{X}) + \log p(\mathbf{X}))^2] \tag{45}$$

$$= \frac{1}{N}(\mathbb{E}[\log^2 p(\mathbf{X})] - \mathrm{h}(\mathbf{X})^2) \tag{46}$$

$$\leq \frac{1}{N} C_1 \tag{47}$$

and apply Chebychev's inequality, showing that with probability greater than $1 - \frac{\delta}{3}$,

$$\left|\mathrm{h}(\mathbf{X}) + \frac{1}{N}\sum_{n=1}^{N} \log p(\mathbf{X}_n)\right| \leq \sqrt{\frac{3C_1}{N\delta}}. \tag{48}$$

The union bound and the triangle inequality applied to (44) and (48) yields the desired result. □

D LIBRARIES USED

For our experiments, we built upon code from the following sources.

- VIBERT (Mahabadi et al., 2021) at `github.com/rabeehk/vibert`.
- TRANSFORMERS (Wolf et al., 2019) at `github.com/huggingface/transformers`.
- DoE (McAllester & Stratos, 2020) at `github.com/karlstratos/doe`.
- SMILE (Song & Ermon, 2019) at `github.com/ermongroup/smile-mi-estimator`.
- InfoNCE, MINE, NWJ, CLUB (Cheng et al., 2020a) at `github.com/Linear95/CLUB`.