Noise-contrastive estimation and related methods

Informatics Research Review

Laszlo Treszkai
(s1765864)

MSc in Artificial Intelligence

School of Informatics

University of Edinburgh

2018

Abstract

Estimating probability distributions is a fundamental problem in machine learning. When the model distributions do not fulfill the normalization constraint, and the partition function is intractable, standard methods like maximum likelihood estimation are not suitable. Gutmann and Hyvärinen (2010) proposed noise-contrastive estimation, which is an estimation principle for statistical models that directly works for unnormalized models, on both discrete and continuous distributions. Their method estimates a probability distribution by performing nonlinear logistic regression to distinguish the data samples from artificially generated noise samples from a reference distribution. Mikolov et al. (2013) modified the objective of this estimator to create word embeddings, and called the resulting method negative sampling. Goodfellow et al. (2014) developed the framework of generative adversarial networks, used for estimating generative models, using an idea similar to that of noise-contrastive estimation, but designed for estimating continuous distributions with neural networks as approximators. This report describes the above three methods, and highlights the relationships between them, including the applicability of each. Their performance is compared against contemporary methods, and against one another where possible, showing that each of them has different strengths, but they all show outstanding results in their own field.

Declaration

I declare that this report was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(Laszlo Treszkai)

Table of Contents

1 Introduction 1

2 Formalizing the problem of estimating unnormalized models. . . . 2

3 Some methods for estimating unnormalized models 3

4 Ideas that led to noise-contrastive estimation. 4

4.1 Contrastive feature learning 4

5 Noise-contrastive estimation 6

5.1 Description of the method 6

5.2 Properties of the estimator 8

5.3 Criticism from the author of this report 10

6 Negative sampling. 11

6.1 NCE on language models 11

7 Generative adversarial nets 13

7.1 Description of GANs 13

7.2 Properties of GANs in comparison with NCE 15

8 Experimental results 16

8.1 Image generation 17

8.2 Language modeling 18

9 Conclusion 19

9.1 Ideas for future work 20

Bibliography 22

1 Introduction

Estimating probability distributions is one of the key problems that machine learning tries to answer: with a good estimator one can do probabilistic reasoning, such as probabilistic inference, classification of data samples, imputation of missing data, or perform simulation-based analyses. The ideal estimator has the following properties: it can model *complex* probability density functions; it allows *fast sampling* from the estimated distribution, and it is *scalable* to high-dimensional data (Rezende et al., 2014). It is often the case that we need to estimate an *unnormalized* model, i.e. one where the outputs of the model distribution do not add up to 1 over the entire state space.

This paper attempts to answer two questions: how, in general, do methods that work by sampling an artificial noise distribution (in addition to the data distribution) perform in contrast to methods that only take samples from the data distribution. Second, to highlight the similarities and differences between noise-contrastive estimation (NCE) and two similar methods, negative sampling and generative adversarial networks (GANs), and to find the domain of applicability of each.

The next section formalizes the problem of estimating unnormalized models. Section 3 describes why maximum likelihood estimation can't solve this problem, and shows how importance sampling can help. Section 4 presents a paper that preceded the conception of NCE, where the same authors used the same idea of sampling a noise distribution to improve learning, but not yet on the general density estimation problem. Then I describe the theory of NCE in section 5, including its first results on generative modeling of natural images. Section 6 describes negative sampling, a variant of NCE that improved the performance of a language model. Section 7 describes generative adversarial networks, and their connection to NCE. Section 8 describes the results of the above methods. Finally, conclusions and ideas for future work are presented in Section 9.

Notation used in this report

This report aims to use a consistent notation for similar functions or variables, which often required changing the letters used in the papers. If a different notation is used in a cited paper, it is given in footnotes when that paper is introduced.

Throughout this report, $r(x) = 1/\big(1 + \exp(-x)\big)$ denotes the logistic sigmoid function. Boldface letters are used for vectors.

2 Formalizing the problem of estimating unnormalized models

There exists a data generating distribution p_{data}, which we want to estimate with a member of the function family $\{p_m^0(\bullet; \boldsymbol{\alpha})\}_{\boldsymbol{\alpha}}$, parametrized by a vector $\boldsymbol{\alpha}$. The samples $\mathbf{x}$ come from a sample space $\mathcal{X}$. When estimating continuous distributions, p_{data} and p_m^0 are probability density functions (*pdf*); when estimating discrete distributions, they are probability mass functions (*pmf*) (Gutmann and Hyvärinen, 2010).

Both a pdf and a pmf can must be normalized, i.e. a pdf f must integrate to 1 over its entire domain, and the values of a pmf g must sum up to 1:

$$\int f(\mathbf{x})\, d\mathbf{x} = 1 \quad \text{for a pdf, or} \quad \sum_{\mathbf{x}} g(\mathbf{x}) = 1 \quad \text{for a pmf.} \tag{1}$$

In case p_m^0 is not necessarily normalized (in other words, when the model is given in *unnormalized* form), we can define the following normalized function, parametrized by $\boldsymbol{\theta}$ (Gutmann and Hyvärinen, 2010):

$$p_m(\mathbf{x}; \boldsymbol{\theta}) = \frac{p_m^0(\mathbf{x}; \boldsymbol{\alpha})}{Z(\boldsymbol{\alpha})}, \qquad Z(\boldsymbol{\alpha}) = \int p_m^0(\mathbf{x}; \boldsymbol{\alpha})\, d\mathbf{x}. \tag{2}$$

$Z(\boldsymbol{\alpha})$ is called the partition function. With many models, for example with

Markov random fields (Roth and Black, 2009), analytically calculating the partition function is not possible (Gutmann and Hyvärinen, 2010), and we need to estimate it when calculating the likelihood of the model. In some domains such as language modelling, where the distributions are discrete and finite, calculating the exact value of the partition function is possible, but computationally inefficient (Mnih and Teh, 2012).

Often we are not interested in the value of the partition function over its entire domain, only for certain arguments: in that case we can just use a constant instead of a function, $c = Z(\boldsymbol{\alpha})$. The full set of parameters that we want to estimate will in this case be $\boldsymbol{\theta} = \boldsymbol{\alpha} \cup \{c\}$.

Furthermore, usually we assume that p_{data} is a member of the model family, i.e. there exists $\boldsymbol{\theta}^\star$ for which $p_m(\bullet; \boldsymbol{\theta}^\star) = p_{\text{data}}(\bullet)$ (Gutmann and Hyvärinen, 2010).

3 Some methods for estimating unnormalized models

Maximum likelihood estimation (MLE) is the standard approach for estimating statistical models (Wasserman, 2004). However, it is not suitable to estimate the normalization constant c: the "likelihood" can be made arbitrarily large by choosing c small (Gutmann and Hyvärinen, 2010). (While the true likelihood is normally defined only for normalized models as the probability of the data under the model, thus has an upper bound of 1.)

In many cases, analytic calculation of $Z(\boldsymbol{\theta})$ is not possible. Numerical calculation of c is inefficient in large discrete distributions (such as language models), and is intractable in high-dimensional continuous spaces (Goodfellow et al., 2016).

Countless methods have been proposed for estimating unnormalized models; Goodfellow et al. (2016, ch. 18) describe many of them in more detail. Below I describe how one of them, importance sampling, can be used for estimating the partition function.

One way to solve this problem is to estimate c by importance sampling (IS), as proposed by Geyer (1992). For IS we need an auxiliary probability distribution f with the same domain as p_m^0, which should be easy to sample from, and whose pdf or pmf can be computed efficiently. We can then estimate the normalizing constant with

$$\hat{c} = \frac{1}{n_y} \sum_{i=1}^{n_y} \frac{p_m^0(\mathbf{y}^{(i)}; \boldsymbol{\theta})}{f(\mathbf{y}^{(i)}}, \tag{3}$$

where $\mathbf{y}^{(1)} \ldots \mathbf{y}^{(n_y)}$ are independent identically distributed samples from f. If $f(\mathbf{y}) > 0$ whenever $p_m^0(\mathbf{y}) > 0$, then by the law of large numbers, the estimator $\hat{c}$ is consistent (Wasserman, 2004). However, the variance of this estimator can be infinite if f has thinner tails than p_m^0. Even if that's not the case, the standard error depends on the number of samples n_y, and how closely f matches p_m^0 (Wasserman, 2004). The advantage of IS is its simplicity, but it has two large drawbacks. First, it requires many samples in order to be accurate, which makes the learning slow. Second, as the model distribution gets closer to p_{data}, it moves farther away from f, thus its variance grows as learning progresses (Mnih and Teh, 2012).

4 Ideas that led to noise-contrastive estimation

4.1 Contrastive feature learning

Gutmann and Hyvärinen (2009) describe a classifier that contrasts whitened natural images with white noise, trained to output 0 for natural images and 1 for the reference data, which method they call contrastive feature learning.

In natural image statistics, the ultimate goal is to learn a generative model of natural images. In an ICA model (Hyvärinen et al., 2004), a natural image vector $\mathbf{x} \in \mathbb{R}^N$ can be written as the linear combination of features $\mathbf{a}_i$ that are as statistically independent as possible:

$$\mathbf{x} = \sum_{i=1}^{M} \mathbf{a}_i s_i, \quad \text{where ideally } \mathbf{a}_i \perp\!\!\!\perp \mathbf{a}_j \text{ for all } i \neq j \tag{4}$$

Learning or calculating the latent variables of an ICA model is computationally intensive (Hyvärinen et al., 2004). This method trains a nonlinear logistic classifier that distinguishes whitened natural images from white noise. For this to work, the classifier needs to identify features beyond the mean and covariance, because these statistics are the same for both classes. The output of the classifier is $r(h(\mathbf{x}))$,[1] where $r(\bullet)$ is the logistic sigmoid, and $h(\mathbf{x}; \boldsymbol{\theta}) = \sum_{m=1}^{M} g(\mathbf{w}_m^T \mathbf{x} + b_m) + \gamma$, with different choices of g. $\boldsymbol{\theta}$ denotes the parameters of the model, which includes the vectors $\mathbf{w}_i$, b_i, γ, and the parameters of g. This results in the following cost function if the number of natural image vectors and reference data are both T:

$$J(\boldsymbol{\theta}) = -\frac{1}{T} \sum_{t=1}^{T} \log \left(r(h(\mathbf{x}^{(i)}; \boldsymbol{\theta})) \right) - \frac{1}{T} \sum_{t=1}^{T} \log \left(1 - r(h(\mathbf{y}^{(i)}; \boldsymbol{\theta})) \right) \tag{5}$$

where $\mathbf{x}^{(i)}$ and $\mathbf{y}^{(i)}$ denote the i-th image vector and noise vector, respectively.

While generative probabilistic models of natural images usually lead to Gabor-like features (i.e. features that are localized, oriented, and indicate bright-dark transitions), this discriminative method also learned Gabor-like features in $\mathbf{w}_i$ (see Figure 1) (Gutmann and Hyvärinen, 2009).

The idea presented in this section, i.e. training a classifier to compare a reference noise distribution and the training dataset, is also what noise-contrastive estimation builds on (Gutmann and Hyvärinen, 2010), but for the task of learning a distribution.

[1]The paper writes $y(\bullet)$ instead of $h(\bullet)$, and $\mathbf{z}^{(i)}$ instead of $\mathbf{y}^{(i)}$.

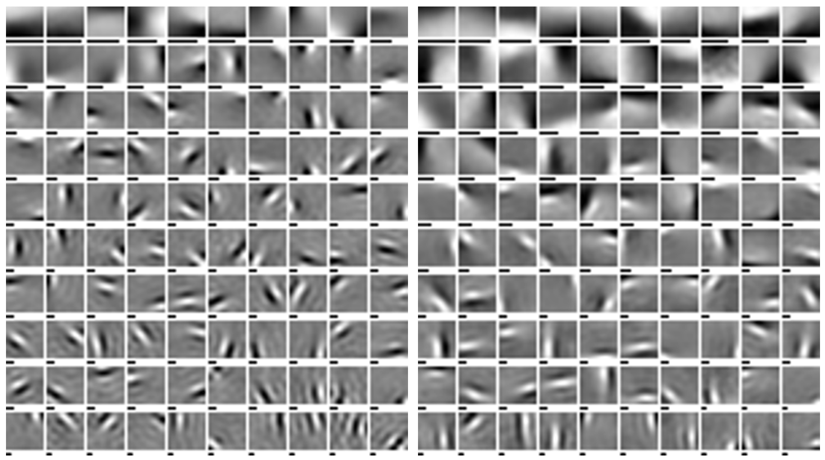

Figure 1: The learned features $\mathbf{w}_i$. Left: features learned with $g(u) = r(u - u_0) + r(-u - u_0)$. Right: $g(u) = \alpha_1[max(0, u - \beta_1)]^{\eta_1} + \alpha_2[max(0, -(u - \beta_2))]^{\eta_2}$. This is Figure 3 of (Gutmann and Hyvärinen, 2009).

Earlier, Hastie et al. (2003, section 14.2.4, pp. 495–497) briefly discussed how density estimation of can be performed by means of logistic regression. The method of Gutmann and Hyvärinen (2010) is able to estimate *unnormalized* models, which was not possible previously, and they proved that their estimator has good statistical properties. This is presented in the next section.

5 Noise-contrastive estimation

5.1 Description of the method

Noise-contrastive estimation (NCE) estimates a parametrized statistical model by performing nonlinear logistic regression to discriminate between the observed data and some artificially generated noise, using the model log-density function in the regression nonlinearity (Gutmann and Hyvärinen, 2010). The authors described a slightly more general version of the estimator in (Gutmann and Hyvärinen, 2012), which is the estimator this section describes.

For this method we need a noise distribution p_n that has similar properties as the one we would need for IS: it is nonzero wherever p_{data} is nonzero, it can be sampled efficiently, and its distribution can be calculated for any input. T_d independent samples $\mathbf{x}^{(i)}$ are taken from p_{data} and $T_n = \nu T_d$ independent samples $\mathbf{y}^{(i)}$ are taken from p_n.[2] A model distribution family p_m is defined, which is chosen according to p_{data}. Notably, the normalization constant of the model can be among the learned parameters (see section 2). Then Gutmann and Hyvärinen (2012) define the following objective function:

$$J_T(\boldsymbol{\theta}) = \frac{1}{T_d}\left[\sum_{i=1}^{T_d} \ln\big(h(\mathbf{x}^{(i)};\boldsymbol{\theta})\big) + \sum_{i=1}^{T_n} \ln\big(1 - h(\mathbf{y}^{(i)};\boldsymbol{\theta})\big)\right], \tag{6}$$

where $T = T_d + T_n$ and

$$h(\mathbf{u};\boldsymbol{\theta}) = r_\nu\big(\ln p_m(\mathbf{u};\boldsymbol{\theta}) - \ln p_n(\mathbf{u})\big). \tag{7}$$

$r_\nu(u) = 1/(1 + \nu \exp(-u))$ is the parametric logistic function, and h is a nonlinear logistic classifier whose nonlinearity is the difference of log-likelihoods of the model and noise distribution (Gutmann and Hyvärinen, 2010). Its output is the probability that $\mathbf{u}$ came from $p_m(\bullet;\boldsymbol{\theta})$ rather than p_n. We can see that $J_T(\boldsymbol{\theta})$ is the weighted average cross-entropy error of this classifier over the data set $\{(\mathbf{x}^{(i)}, 0)\}_{i=1}^{T_d} \cup \{(\mathbf{y}^{(i)}, 1)\}_{i=1}^{T_n}$, with bigger weights on the noise samples (Bishop, 1995).

The parameter $\hat{\boldsymbol{\theta}}_T$ that maximizes J_T is found iteratively with a gradient-based optimizer; the authors used the nonlinear conjugate gradient algorithm of Rasmussen (2006) (Gutmann and Hyvärinen, 2010, 2012). Algo-

[2]Gutmann and Hyvärinen (2010) stated the theorems for the case where $T_d = T_n$, i.e. $\nu = 1$, in their first publication on NCE.

rithm 1 summarizes the training procedure of NCE, and figure 2 visualizes the algorithm on a simple model.

Algorithm 1 Minibatch stochastic gradient descent training of a noise-comparative estimator (Gutmann and Hyvärinen, 2012).

```
for number of training iterations do
    • Sample minibatch $\{\mathbf{x}^{(1)}, \dots, \mathbf{x}^{(T_d)}\}$ from data generating distribution $p_d$
    • Sample minibatch $\{\mathbf{y}^{(1)}, \dots, \mathbf{y}^{(T_n)}\}$ from noise distribution $p_n$
    • Update the parameters of $p_m$ by ascending the stochastic gradient of $J_T$,
        $$\nabla_{\boldsymbol{\theta}} \frac{1}{T_d} \Big[ \sum_{i=1}^{T_d} \ln \big( h(\mathbf{x}^{(i)}; \boldsymbol{\theta}) \big) + \sum_{i=1}^{T_n} \ln \big( 1 - h(\mathbf{y}^{(i)}; \boldsymbol{\theta}) \big) \Big].$$
end for
```

5.2 Properties of the estimator

As T_d increases, the objective function J_T converges in probability to J:

$$J(\boldsymbol{\theta}) = \mathbb{E}_{\mathbf{x} \sim p_d} \big[\ln \big(h(\mathbf{x}; \boldsymbol{\theta}) \big) \big] + \nu \mathbb{E}_{\mathbf{y} \sim p_n} \big[\ln \big(1 - h(\mathbf{y}; \boldsymbol{\theta}) \big) \big]. \tag{8}$$

(Gutmann and Hyvärinen, 2012) prove that we can learn the true distribution in the non-parametric limit and in the limit of infinite data, i.e. that J attains its maximum if and only if $p_d(\bullet) = p_m(\bullet; \boldsymbol{\theta}^\star)$ for all inputs. Remember that it is allowed to have the normalization constant to be in $\boldsymbol{\theta}$, meaning this property remains true for unnormalized models.

The authors also prove that if certain weak conditions are fulfilled by p_n and p_m, then $\hat{\boldsymbol{\theta}}_T$ converges in probability to $\boldsymbol{\theta}^\star$, i.e. $\hat{\boldsymbol{\theta}}_T$ is a consistent estimator.

Gutmann and Hyvärinen (2010) also discuss the choice of noise distribution p_n. Obviously it should be easy to sample from and easily evaluated so that Algorithm 1 can be executed efficiently. p_n should also fulfill the conditions that provably lead to a consistent estimator, but these are relatively weak criteria, according to (Gutmann and Hyvärinen, 2010). Gutmann and Hyvärinen (2010) propose a Gaussian, uniform, a Gaussian mixture distribution, or an ICA distribution as a choice of p_n. Also, as Gutmann and Hyvärinen (2010) put it, the "noise distribution should ideally be close to the data distribution" [in order for the mean squared error (MSE) of the estimator to be low] – and this idea is the foundation of general adversarial networks in (Goodfellow et al., 2014), which are discussed in section 7 of this report. Mnih and Teh (2012) reported notable differences in performance with discrete distributions between using a uniform and a problem-specific noise distribution.

As we stated earlier, the normalizing constant c can be a member of $\boldsymbol{\theta}$, to estimate unnormalized models. Mnih and Teh (2012) found that if the family of unnormalized functions $\{p_m(\bullet; \boldsymbol{\theta})\}_{\boldsymbol{\theta}}$ is given enough capacity, i.e. it is a family with enough parameters to represent a broad class of functions, then the network self-normalizes without an additional normalizing constant.

However, if the normalizing constant is a parameter of the model, then one can use the estimated constant c for model comparison, as stated by Pihlaja et al. (2010).

The fact that the method cannot be used for models with latent variables (Gutmann and Hyvärinen, 2012, Sec. 5) prevents its use with deep belief networks or directed Boltzmann machines, and restricts its usability to fully visible models.

Gutmann and Hirayama (2012) proved that NCE can be used to estimate discrete models, and it still remains a consistent estimator – a property that let this method stand the test of time on language models, as we'll see in the next section. Gutmann and Hirayama (2012) reported that when estimating a fully visible Boltzmann machine, the error of NCE was similar to that of maximum pseudo-likelihood.

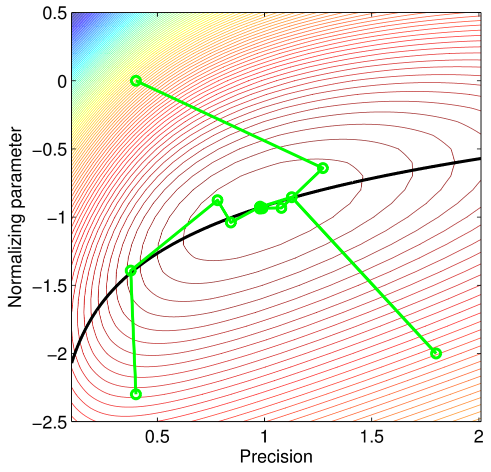

Figure 2: Contour map of the objective function J_T for the estimation of the precision and normalizing constant of a univariate Gaussian model, $\log p_m(x) = -\alpha\, x^2/2 + c$. Black line: parameter combinations $\{\alpha, c\}$ that make the model normalized, i.e. $c = -\log\alpha - \log(2\pi)/2$. Green lines: different trajectories of the estimated parameters during the optimization process. We can see how both parameters converge to the true values, $\alpha = 1$, $c \approx -0.92$. Reproduced from (Gutmann and Hyvärinen, 2013).

5.3 Criticism from the author of this report

Gutmann and Hyvärinen (2010) mention that the optimizer may get stuck in local optima (which is a typical problem in non-convex optimization), but it doesn't mention the variance of the MSE with different random initializations. Gutmann and Hyvärinen (2012) address this and report confidence intervals for all methods instead of just reporting the mean.

Although the proofs were not provided in (Gutmann and Hyvärinen, 2010), the Gutmann and Hyvärinen (2012) addresses this problem. Their work does not display methodological flaws, and they do not exaggerate their results: the obtained weights are indeed Gabor-filters, and the generated image patches show clear boundaries. A side-by-side comparison of the image patches generated by NCE and a state-of-the-art contemporary method would have provided more support for their work.

6 Negative sampling

6.1 NCE on language models

NCE was first used by probabilistic language modelling by Mnih and Teh (2012), where they learned to predict the next word in a sentence, given the previous 2 words. Mnih and Teh (2012) used a unigram noise distribution, i.e. one where $p_n(w)$ equals the estimated probability of finding the word w in any position in the sentence. (This distribution achieved much better performance than a uniform distribution, showing that the choice of p_n can make a big difference.) They needed 25 noise samples for every real sample in order to achieve the same perplexity scores as maximum likelihood; however, training with NCE resulted in a 45-fold speedup.

Description of negative sampling

The Skip-gram model developed by Mikolov et al. (2013a) is a language model that maps word phrases to continuous vectors. (These are the so-called *vector embedding* of a word.) Ideally, these vectors would share the same similarities as the words: v_{France} (the vector embedding of "France") would be close to v_{Italy}, and $v_{\text{small}} + (v_{\text{bigger}} - v_{\text{big}})$ would be close to v_{smaller}.

In comparison to the previous n-gram CBOW model (which predicted maximized the probability of finding a word given the previous words), Skip-gram tries to maximize the probability of finding a word in the presence of a word *anywhere* in its c-radius neighborhood. Formally, given a sequence of training words i.e. maximize the following average probability

(Mikolov et al., 2013a):

$$\frac{1}{T}\sum_{t=1}^{T}\sum_{-c\leq j\leq c, j\neq 0}\log p(w_{t+j}|w_t) \tag{9}$$

where c is the size of the training context. The conditional probability $p(w_{t+j}|w_t)$ is traditionally estimated using the softmax function (Mikolov et al., 2013b):

$$p(w_O|w_I) = \frac{\exp\left({v'_{w_O}}^\top v_{w_I}\right)}{\sum_{w\in\mathcal{W}}\exp\left({v'_w}^\top v_{w_I}\right)}, \tag{10}$$

where v_w and v'_w are the "input" and "output" vector representations of the word $w \in \mathcal{W}$, with $\mathcal{W}$ denoting the word vocabulary.

Calculating the denominator in the above equation is computationally inefficient because it requires looping through the entire vocabulary. One way to solve this problem is using the hierarchical softmax (Morin and Bengio, 2005), but the description of that method falls outside the scope of this document.

Another solution is to use NCE as in (Mnih and Teh, 2012). But as the goal of a Skip-gram model is to learn vector representations of words that have the aforementioned useful properties, NCE can be simplified further, as long as it maximizes the dot product of input and output vector of related words, i.e. ${v'_{w_O}}^\top v_{w_I}$ (Mnih and Teh, 2012). (Neither Mikolov et al. (2013a), nor Mikolov et al. (2013b) provided a theoretical argument why this should be the case.) Mikolov et al. (2013b) thus replaced the conditional probability

$\log p(w_O|w_I)$ in equation 8 with the following objective function[3]:

$$\log r\big({v'_{w_O}}^\top v_{w_I}\big) + \sum_{i=1}^{k} \mathbb{E}_{w_i \sim p_n(w)} \Big[\log \Big(1 - r\big({v'_{w_i}}^\top v_{w_I}\big)\Big)\Big] \tag{11}$$

where r is again the sigmoid function. The reader can compare this equation with the objective J_T of equation 6. With k noise samples for every true pair ($\nu = T_n/T_d$ in equation 6), and replacing the dot product $({v'_{w_O}}^\top v_{w_I})$ for the difference in log-likelihood $(\log p_m(\mathbf{u}; \boldsymbol{\theta}) - \log p_n(\mathbf{u}))$, the relation is clear. Mikolov et al. (2013b) called this new method *negative sampling*. The idea of training a logistic classifier to contrast data points with noise samples in order to find useful features is the same. But while in (Gutmann and Hyvärinen, 2010) these features were the parameters of a statistical model, in (Mikolov et al., 2013b) the features were the coordinates of word embeddings.

The system of Mikolov et al. (2013b) resulted in word embeddings with more interesting qualities than previous systems. Further results are provided in section 8.

As we see, negative sampling is not suitable for estimating arbitrary probability distributions, as it was specifically designed to create word embeddings.

7 Generative adversarial nets

7.1 Description of GANs

The method of generative adversarial networks (GAN) was first described in (Goodfellow et al., 2014) for estimating generative models.

A GAN trains two models simultaneously: a generative model G and a

[3]Mikolov et al. (2013b) used $\sigma(\bullet)$ instead of $r(\bullet)$, and P_n instead of p_n. Also, here the equality $r(-x) = 1 - r(x)$ was used.

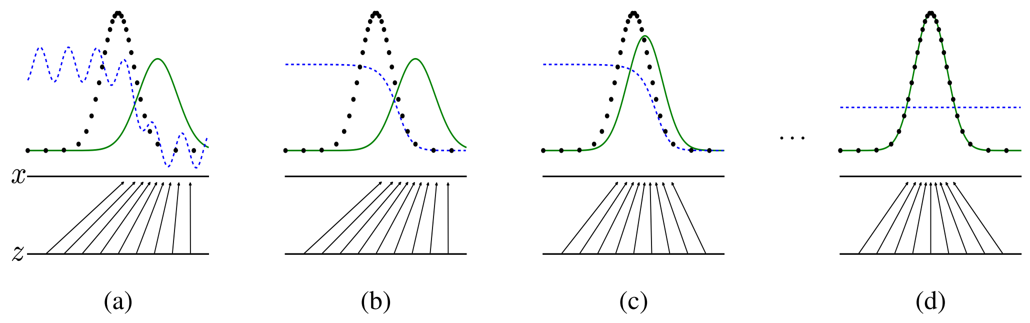

Figure 3: Training process of a GAN. Black horizontal lines: domain of noise (bottom) and data (top) distributions. Arrows between the two represent the mapping G. Black dotted curve: p_{data}. Green curve: p_g, the distribution of $G(\mathbf{y})$. Blue dotted curve: output of D. (a) Random initialization. (b) D is updated to better distinguish $\mathbf{x}$ from $G(\mathbf{y})$, while G is held fixed. (c) G is updated to match p_{data} better. (d) In the limit, if G and D have enough capacity, $p_g = p_{\text{data}}$ and $D(\bullet) = 1/2$ everywhere. Figure reproduced from (Goodfellow et al., 2014).

discriminative model D. (Figure 3 illustrates the process.) The input of G is a sample $\mathbf{y}$ of a noise distribution p_n,[4] and its output is in the domain of p_{data}. The input of D is either the output of the generative model, $G(\mathbf{y})$, or a sample from the data generating distribution $\mathbf{x} \sim p_{\text{data}}$, and its output is in $(0, 1)$, representing the probability that its input came from the data rather than the generator. These two models are trained simultaneously on the same objective function V, but D is trained to maximize V, i.e. minimize the error of its predictions, while G is trained to minimize V; thereby the training of GANs can be modelled by playing a two-player minimax game.

The objective function V is the negative binary cross-entropy error of the classifier D:

$$V(D, G) = \mathbb{E}_{\mathbf{x}\sim p_{\text{data}}}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{y}\sim p_n}[\log\left(1 - D(G(\mathbf{y}))\right)]. \tag{12}$$

[4]Goodfellow et al. (2014) uses $\mathbf{z} \sim p_{\mathbf{z}}$ instead of $\mathbf{y} \sim p_n$.

Algorithm 2 Minibatch stochastic gradient descent training of a generative adversarial net (Goodfellow et al., 2014). (The inner loop of multiple updates on $\boldsymbol{\theta}_d$ is omitted here, as the authors used only one iteration.) Note that the original paper has an incorrect subscript for the noise prior.

```
for number of training iterations do
    • Sample minibatch $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(T)}\}$ from $p_{\text{data}}$
    • Sample minibatch $\{\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(T)}\}$ from noise distribution $p_n$
    • Update the parameters of $D$ by ascending the stochastic gradient,
        $$\nabla_{\boldsymbol{\theta}_d} \frac{1}{T} \sum_{i=1}^{T} \left[ \log \left( D(\mathbf{x}^{(i)}) \right) + \log \left( 1 - D(G(\mathbf{y}^{(i)})) \right) \right].$$
    • Resample minibatch $\{\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(T)}\}$ from noise distribution $p_n$
    • Update the parameters of $G$ by descending the stochastic gradient,
        $$\nabla_{\boldsymbol{\theta}_g} \frac{1}{T} \sum_{i=1}^{T} \log \left( 1 - D(G(\mathbf{y}^{(i)})) \right).$$
end for
```

7.2 Properties of GANs in comparison with NCE

Algorithm 2 describes the training procedure of a GAN. It looks similar to the training of NCE in Algorithm 1, and indeed, if we fix the function G to be the identity function (i.e. $G(\mathbf{y}) = \mathbf{y}$ for all $\mathbf{y}$), and substitute $r\big(\log p_m(\mathbf{x}; \boldsymbol{\theta}) - \log p_n(\mathbf{x})\big)$ for D in Equation 12, we get the same objective function as J in Equation 8, up to a constant factor of 2. This also means that while NCE works by learning in the discriminator, GANs work by learning in the generator (Goodfellow et al., 2014).

Goodfellow (2014) claims that this is an important distinction, as it makes the gradients of the objective functions quite different. GANs are asymptotically consistent (Goodfellow et al., 2014), just like MLE or NCE, meaning that in the limit of infinite data they reach a stationary point with the model parameters, one which defines a model distribution that matches the data distribution almost everywhere. However, for MLE and NCE this stationary point is the maximum of the objective function, while for GANs it is a saddle point, which is a local minimum for the generator and a local maximum for the discriminator.

Furthermore, according to Goodfellow (2017), GANs focus on sample generation, not on explicitly expressing the model distribution p_m. (Although the paper mentions the existance of GANs that can do both, no examples are provided.) In contrast, as we saw in section 5, NCE can directly calculate p_m for any input, which is required for Bayesian inference (Murphy, 2012).

On the other hand, GANs fix the problem that learning with NCE slows rapidly as p_m approaches p_{data} because of the fixed noise distribution (Goodfellow et al., 2014). Unfortunately, the author does not make a quantitative comparison between the two methods, nor are any learning curves provided for GANs.

According to (Goodfellow et al., 2014), both of these parametric functions D and G are typically calculated by deep neural networks, hence the name of the method, *"generative adversarial networks"*.

A great limitation of GANs is that the generator function G must be differentiable, so GANs cannot estimate discrete distributions (Goodfellow et al., 2014); Goodfellow (2017) state three ways of how this could be solved. As we saw in section 5, this is not the case with NCE, which is one of the reasons why that technique is frequently used in language modelling, despite the popularity of GANs.

8 Experimental results

While the methods presented in this report are related theoretically, they were designed for different purposes, and rarely used on the same problem

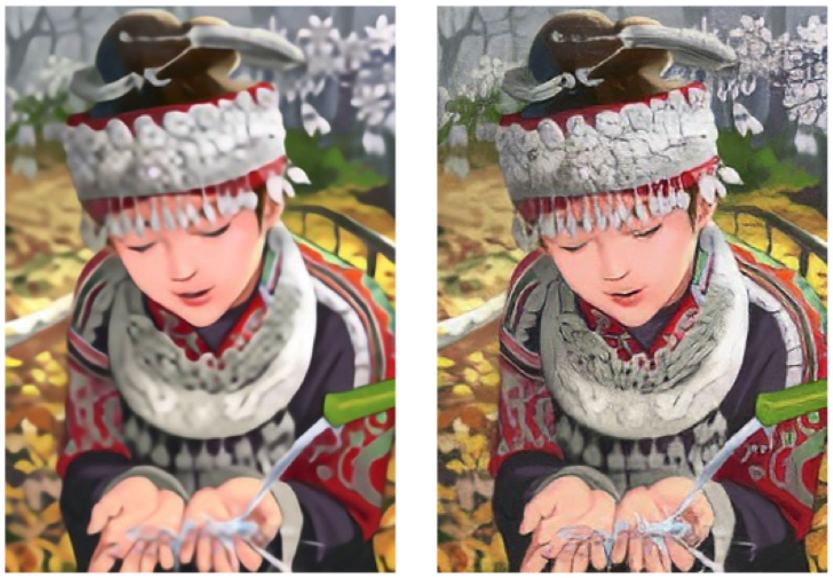

Figure 4: Output of SRResNet (left) and SRGAN (right), from a low-resolution image. The right one shows much clearer details. Figure reproduced from (Ledig et al., 2016).

and same dataset. As such, direct comparison is often not possible, and they are compared against other methods of the time which do not use noise samples in the training process.

8.1 Image generation

A few of the possible methods for comparing generative methods are manually measuring the fidelity of the generated samples (e.g. the visual fidelity in case of images), and estimating the log-likelihood of the models under the same data set (Theis et al., 2015). Unfortunately, these two properties do not come hand in hand: it is possible to have a model that has high likelihood and low fidelity, or the other way round (Theis et al., 2015).

GANs perform very well in a number of image generation tasks, for example generating superresolution images (figure 4) (Ledig et al., 2016) or generating images from textual description (Nguyen et al., 2016). All of these works show images that have higher visual fidelity than competing works.

8.2 Language modeling

In the field of language modeling, the comparison is *slightly* easier, although usage of proprietary data sets and uneven allocation of computation resources make it hard to tell objectively which method is the best.

The vector embeddings created by the Skip-gram model of Mikolov et al. (2013b) show the property of *compositionality*, i.e. where the closest vector to the element-wise sum of v_{Vietnam} and v_{capital} is v_{Hanoi} (the vector embedding of the capital of Vietnam). With earlier methods this was not possible (Mikolov et al., 2013b).

Mikolov et al. (2013b) quantitatively compare the accuracy of their 1000-dimensional Skip-gram model with different embedding methods on an analogical reasoning task. (The task was developed by Google Inc., the employer of the authors.) They find that under equal training times, NCE performs better than the hierarchical softmax on all measures of accuracy, while their negative sampling performs even better. All three had far better accuracy than the baseline CBOW model.

Mnih and Kavukcuoglu (2013) performed the same analogical reasoning task, both with their own model vLBL (which used NCE) and with the Skip-gram model of Mikolov et al. (2013b). They could not reach the accuracy of the 1000-dimensional Skip-gram model stated above, but they could not use the same training set, and training their model used 75 times fewer CPU cycles than the 1000-dimensional Skip-gram model. The reported performance of the 300-dimensional vLBL was significantly better than the 300-D Skip-gram model (64% vs. 55%) on a similarity task, and both of these far outperfromed the 300-D CBOW model (36%).

A number of recent systems use NCE for model estimation in natural language processing (NLP) tasks, such as machine translation (Mi et al., 2016), speech recognition (Xiong et al., 2016), or information retrieval (Sordoni et al., 2015), often without mentioning the details of the method, suggesting that it has become standard in the field of NLP. As a final example: Chen et al. (2015) compared NCE with the traditional objective of cross-entropy minimization (CE) in a speech recognition task with recurrent neural networks. In their experiments NCE reached the same perplexity scores as CE (and they were 21 percent lower than a competing 5-gram model), but NCE required half the training time as CE; the training time of NCE was independent of output vocabulary size, while that of CE increased with it; and the testing speed of NCE was 56 times higher.

9 Conclusion

This report presented the problems with the intractable partition function, and three solutions for it. Noise-contrastive estimation (Gutmann and Hyvärinen, 2010) is a general method to estimate continuous or probability distributions of fully visible models (see section 5). Negative sampling (Mikolov et al., 2013a) is a competing method specifically developed to create vector embeddings of words or word phrases, as a special case of NCE (see section 6). Generative adversarial networks (Goodfellow et al., 2014) represent a family of methods that focus on sample generation of continuous distributions (see section 7).

We saw that while the theoretical foundations of these methods are similar, they are employed in different domains (see section 8). GANs are limited to continuous domains, and so are primarily used in the generation of photorealistic images, with good results. Negative sampling is used solely for efficiently creating word embeddings that preserve the similarity of the word phrases, which vectors can then be used for natural language tasks. NCE is applied on discrete problems, with outstanding results on language models, but GANs seem to have completely replaced it in the continuous case.

While the race between NCE and negative sampling is not yet settled, all three of these methods perform better than the alternatives in the aforementioned tasks. Negative sampling produces quality word embeddings, and NCE and GANs both fulfill the requirements of an ideal estimator stated in the introduction: they can model complex distributions, they allow fast sampling, and they are scalable to high-dimensional data. This shows the strength of the idea of Gutmann and Hyvärinen (2010).

9.1 Ideas for future work

Mikolov et al. (2013b) used negative sampling to create word embeddings, thereby performing dimensionality reduction from a 100,000-dimensional discrete space to a 1000-dimensional continuous space, while retaining certain qualities of the original space, such as word similarity. A similar method might be potentially applied in other domains, for example customer analysis, to create a mapping from the preferences of a user to a lower-dimensional continuous vector. A user could be represented as a binary vector, with each coordinate standing for a different product, having the value of 1 if the product is found in the purchase history. This might improve currently existing marketing analysis techniques, such as (Wedel and Kannan, 2016).

Gutmann and Hyvärinen (2010) suggested that the noise distribution should ideally be close to p_{data}, which Goodfellow (2014) took to the extreme case of updating p_n in every iteration to the last learned p_m, and called this method *self-contrastive estimation*. That work did not attempt to evaluate whether the good properties of NCE still hold in this new setting, nor did it perform experiments about performance. As it could potentially provide steeper gradients for the discriminator even in the later stages of training, this area might be a fruitful research for further improving the training speed and accuracy of NCE-based language methods.

Bibliography

Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford university press.

Chen, X., Liu, X., Gales, M. J., and Woodland, P. C. (2015). Recurrent neural network language model training with noise contrastive estimation for speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2015 IEEE International Conference on*, pages 5411–5415. IEEE.

Geyer, C. J. (1992). On the convergence of monte carlo maximum likelihood calculations. *Journal of the Royal Statistical Society B*, 56:261–274.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

Goodfellow, I. J. (2014). On distinguishability criteria for estimating generative models. *arXiv preprint arXiv:1412.6515*.

Goodfellow, I. J. (2017). NIPS 2016 tutorial: Generative adversarial networks. *CoRR*, abs/1701.00160.

Goodfellow, I. J., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative Adversarial Networks. *ArXiv e-prints*.

Gutmann, M. and Hirayama, J.-i. (2012). Bregman divergence as general framework to estimate unnormalized statistical models. *arXiv preprint arXiv:1202.3727*.

Gutmann, M. and Hyvärinen, A. (2009). Learning features by contrasting natural images with noise. *Artificial Neural Networks–ICANN 2009*, pages 623–632.

Gutmann, M. and Hyvärinen, A. (2010). Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 297–304.

Gutmann, M. U. and Hyvärinen, A. (2012). Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13(Feb):307–361.

Gutmann, M. U. and Hyvärinen, A. (2013). Estimation of unnormalized statistical models without numerical integration. In *Proc. Workshop on Information Theoretic Methods in Science and Engineering (WITMSE2013)*.

Hastie, T., Tibshirani, R., and Friedman, J. (2003). The elements of statistical learning, corrected ed. *Berlin: Springer. Haxby, JV, Gobbini, MI, Furey, ML, Ishai, A., Schouten, JL, & Pietrini, P.(2001). Distributed and overlapping representations of faces and objects in ventral temporal cortex. Science*, 293(5539):24252430.

Hyvärinen, A., Karhunen, J., and Oja, E. (2004). *Independent component analysis*, volume 46. John Wiley & Sons.

Ledig, C., Theis, L., Huszar, F., Caballero, J., Aitken, A. P., Tejani, A., Totz, J., Wang, Z., and Shi, W. (2016). Photo-realistic single image super-resolution using a generative adversarial network. *CoRR*, abs/1609.04802.

Mi, H., Wang, Z., and Ittycheriah, A. (2016). Vocabulary manipulation for neural machine translation. *CoRR*, abs/1605.03209.

Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013a). Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013b). Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Mnih, A. and Kavukcuoglu, K. (2013). Learning word embeddings efficiently with noise-contrastive estimation. In *Advances in neural information processing systems*, pages 2265–2273.

Mnih, A. and Teh, Y. W. (2012). A fast and simple algorithm for training neural probabilistic language models. *arXiv preprint arXiv:1206.6426*.

Morin, F. and Bengio, Y. (2005). Hierarchical probabilistic neural network language model. In *Aistats*, volume 5, pages 246–252.

Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. MIT press.

Nguyen, A., Yosinski, J., Bengio, Y., Dosovitskiy, A., and Clune, J. (2016). Plug & play generative networks: Conditional iterative generation of images in latent space. *CoRR*, abs/1612.00005.

Pihlaja, M., Gutmann, M., and Hyvarinen, A. (2010). A family of computationally efficient and simple estimators for unnormalized statistical models. *arXiv preprint arXiv:1203.3506*.

Rasmussen, C. (2006). Conjugate gradient algorithm. Technical report, version 2006-09-08.

Rezende, D. J., Mohamed, S., and Wierstra, D. (2014). Stochastic backpropagation and approximate inference in deep generative models. *arXiv preprint arXiv:1401.4082*.

Roth, S. and Black, M. J. (2009). Fields of experts. *International Journal of Computer Vision*, 82(2):205–229.

Sordoni, A., Galley, M., Auli, M., Brockett, C., Ji, Y., Mitchell, M., Nie, J., Gao, J., and Dolan, B. (2015). A neural network approach to context-sensitive generation of conversational responses. *CoRR*, abs/1506.06714.

Theis, L., Oord, A. v. d., and Bethge, M. (2015). A note on the evaluation of generative models. *arXiv preprint arXiv:1511.01844*.

Wasserman, L. (2004). *All of Statistics: A Concise Course in Statistical Inference*. Springer Texts in Statistics. Springer.

Wedel, M. and Kannan, P. (2016). Marketing analytics for data-rich environments. *Journal of Marketing*, 80(6):97–121.

Xiong, W., Droppo, J., Huang, X., Seide, F., Seltzer, M., Stolcke, A., Yu, D., and Zweig, G. (2016). Achieving human parity in conversational speech recognition. *CoRR*, abs/1610.05256.