Maximum Reconstruction Estimation for Generative Latent-Variable Models

Yong Cheng[#], Yang Liu[†] [*] and Wei Xu[#]
[#]Institute for Interdisciplinary Information Sciences, Tsinghua University, Beijing, China
[†]State Key Laboratory of Intelligent Technology and Systems
Tsinghua National Laboratory for Information Science and Technology
Department of Computer Science and Technology, Tsinghua University, Beijing, China
chengyong3001@gmail.com
{liuyang2011, weixu}@tsinghua.edu.cn

Abstract

Generative latent-variable models are important for natural language processing due to their capability of providing compact representations of data. As conventional maximum likelihood estimation (MLE) is prone to focus on explaining irrelevant but common correlations in data, we apply *maximum reconstruction estimation* (MRE) to learning generative latent-variable models alternatively, which aims to find model parameters that maximize the probability of reconstructing the observed data. We develop tractable algorithms to directly learn hidden Markov models and IBM translation models using the MRE criterion, without the need to introduce a separate reconstruction model to facilitate efficient inference. Experiments on unsupervised part-of-speech induction and unsupervised word alignment show that our approach enables generative latent-variable models to better discover intended correlations in data and outperforms maximum likelihood estimators significantly.

Introduction

The need to learn latent structures from unlabeled data arises in many different problems in natural language processing (NLP), including part-of-speech (POS) induction (Merialdo 1994; Johnson 2007), word alignment (Brown et al. 1993; Vogel, Ney, and Tillmann 1996), syntactic parsing (Klein and Manning 2004; Smith and Eisner 2005), and semantic parsing (Poon and Domingos 2009). Generative latent-variable models such as hidden Markov models (HMMs) and latent-variable probabilistic context-free grammars (L-PCFGs) have been widely used for unsupervised structured prediction due to their capability of providing compact representations of data.

Maximum likelihood estimation (MLE) is the standard criterion for training generative latent-variable models: maximizing the likelihood of observed data by marginalizing over latent variables, typically via the Expectation Maximization (EM) algorithm. However, recent studies have revealed that MLE suffers from a significant problem: it may guide the model to focus on explaining irrelevant but common correlations in the data (Ganchev et al. 2010). For example, in unsupervised POS induction, estimating tag probabilities are often prone to be disrupted by high-frequency words such as "," and "is" (see Table 3).

In this work, we introduce *maximum reconstruction estimation* (MRE) (Hinton, Osindero, and Teh 2006; Vincent et al. 2008; Bengio 2009; Vincent et al. 2010; Socher et al. 2011; Ammar, Dyer, and Smith 2014; Alain et al. 2015) for learning generative latent-variable models. The basic idea is to circumvent irrelevant but common correlations by maximizing the probability of reconstructing observed data. The new objective is based on an *encoding-reconstruction* framework: first generating a latent structure conditioned on the observed data (*encoding*) and then re-generating the observation based on the latent structure (*reconstruction*). Our approach has the following advantages:

1. *Direct learning of model parameters*. As we are only interested in learning the parameters of *data distribution* (i.e., $P(\mathbf{x};\boldsymbol{\theta})$), it is challenging to define the *reconstruction distribution* (i.e., $P(\mathbf{x}|\mathbf{x};\boldsymbol{\theta})$) using data distribution parameters (i.e., $\boldsymbol{\theta}$) due to the intricate dependencies between sub-structures in generative latent-variable models. While previous work has to maintain separate sets of model parameters for encoding and reconstruction to facilitate efficient inference (Ammar, Dyer, and Smith 2014), our approach is capable of directly learning intended model parameters.

2. *Tractable inference*. Our approach keeps the same tractability of learning generative latent-variable models as its MLE counterpart. In this work, we apply MRE to learning two classic generative latent-variable models (i.e., HMMs and IBM translation models) and design efficient dynamic programming algorithms to calculate expectations exactly.

We evaluate our approach on unsupervised POS induction and unsupervised word alignment. Experiments show that MRE enables generative latent-variable models to better discover intended correlations in data and thus leads to significant improvements over MLE.

[*] Yang Liu is the corresponding author: liuyang2011@ tsinghua. edu.cn.

Maximum Reconstruction Estimation

Let $\mathbf{x}$ be an observation, $\mathbf{z}$ be a latent structure, and $P(\mathbf{x};\boldsymbol{\theta})$ be a generative latent-variable model parameterized by $\boldsymbol{\theta}$:

$$P(\mathbf{x};\boldsymbol{\theta}) = \sum_{\mathbf{z}} P(\mathbf{x},\mathbf{z};\boldsymbol{\theta}) \tag{1}$$

Given a set of training examples $D = \{\mathbf{x}^{(s)}\}_{s=1}^{S}$, the standard *maximum likelihood estimation* (MLE) criterion aims to find a set of parameters that maximizes the log-likelihood of the training data:

$$\boldsymbol{\theta}^*_{\text{MLE}} = \operatorname*{argmax}_{\boldsymbol{\theta}} \left\{ \sum_{s=1}^{S} \log P(\mathbf{x}^{(s)};\boldsymbol{\theta}) \right\} \tag{2}$$

After learning model parameters, the Viterbi test-time predictions can be calculated as follows:

$$\mathbf{z}^*_{\text{MLE}} = \operatorname*{argmax}_{\mathbf{z}} \Big\{ P(\mathbf{x},\mathbf{z};\boldsymbol{\theta}) \Big\} \tag{3}$$

Although MLE has been widely used in learning latent variable models, Ganchev et al. (2010) indicate that it may guide the model to focus on explaining irrelevant but common (rather than intended) correlations in data.

Alternatively, we are interested in modeling the probability of reconstructing observations via latent structures:

$$P(\hat{\mathbf{x}}|\mathbf{x};\boldsymbol{\theta}) = \sum_{\mathbf{z}} \underbrace{P(\mathbf{z}|\mathbf{x};\boldsymbol{\theta})}_{\textit{encoding}} \underbrace{P(\hat{\mathbf{x}}|\mathbf{x},\mathbf{z};\boldsymbol{\theta})}_{\textit{reconstruction}} \tag{4}$$

$$= \mathbb{E}_{\mathbf{z}|\mathbf{x};\boldsymbol{\theta}} \Big[P(\hat{\mathbf{x}}|\mathbf{x},\mathbf{z};\boldsymbol{\theta}) \Big] \tag{5}$$

where $P(\mathbf{z}|\mathbf{x};\boldsymbol{\theta})$ is an *encoding* sub-model that generates a latent structure $\mathbf{z}$ given the observation $\mathbf{x}$ and $P(\hat{\mathbf{x}}|\mathbf{x},\mathbf{z};\boldsymbol{\theta})$ is a *reconstruction* sub-model that reconstructs the observation $\hat{\mathbf{x}}$ given the just generated latent structure $\mathbf{z}$ as well as the observation itself. Often, we ignore the dependency on the observation itself and simply write the reconstruction sub-model as $P(\hat{\mathbf{x}}|\mathbf{z};\boldsymbol{\theta})$. Note that unlike Ammar et al. (2014), we require that both the encoding and reconstruction sub-models share the original model parameters to be learned. This forces training algorithms to focus on learning intended models.

As a result, *maximum reconstruction estimation* (Hinton, Osindero, and Teh 2006; Vincent et al. 2008; Bengio 2009; Vincent et al. 2010; Socher et al. 2011; Ammar, Dyer, and Smith 2014; Alain et al. 2015) aims to find a set of parameters that maximize the product of reconstruction probabilities of the training data:

$$\boldsymbol{\theta}^*_{\text{MRE}} = \operatorname*{argmax}_{\boldsymbol{\theta}} \left\{ \sum_{s=1}^{S} \log P(\hat{\mathbf{x}}^{(s)}|\mathbf{x}^{(s)};\boldsymbol{\theta}) \right\} \tag{6}$$

We use the exponentiated gradient algorithm (Kivinen and Warmuth 1997) with adaptive learning rate (Bagos, Liakopoulos, and Hamodrakas 2004) for estimating model parameters. The decision rule for computing Viterbi test-time predictions is given by

$$\mathbf{z}^*_{\text{MRE}} = \operatorname*{argmax}_{\mathbf{z}} \Big\{ P(\hat{\mathbf{x}}|\mathbf{z};\boldsymbol{\theta}) P(\mathbf{z}|\mathbf{x};\boldsymbol{\theta}) \Big\} \tag{7}$$

We aim to reconstruct $\mathbf{x}$ in the following sections which indicates $\hat{\mathbf{x}} = \mathbf{x}$. For simplicity, we do not differentiate $\hat{\mathbf{x}}$ and $\mathbf{x}$ explicitly, and donate $\hat{\mathbf{x}}$ as $\mathbf{x}$.

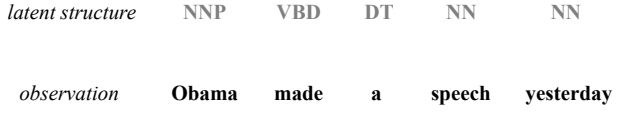

Figure 1: Part-of-speech induction. Given an observed English sentence, the task is to induce the latent sequence of part-of-speech tags.

Applications to Generative Latent-Variable Models

We apply the maximum reconstruction criterion to training two classical generative latent-variable models: hidden Markov models for unsupervised POS induction (Section 3.1) and IBM translation models for unsupervised word alignment (Section 3.2).

Hidden Markov Models for Unsupervised POS Induction

Hidden Markov models (HMMs) are a widely used statistical tool for modeling latent sequential structures. Figure 1 shows an example of unsupervised part-of-speech induction. Given a natural language sentence "Obama made a speech yesterday", HMMs can be used to infer the latent sequence of part-of-speech tags "NNP VBD DT NN NN".

More formally, let $\mathbf{x} = \mathbf{x}_1,\ldots,\mathbf{x}_n,\ldots,\mathbf{x}_N$ be an observed natural language sentence with N words and $\mathbf{z} = \mathbf{z}_1,\ldots,\mathbf{z}_n,\ldots,\mathbf{z}_N$ be the corresponding sequence of part-of-speech tags. We use x and z to denote a single word and tag, respectively. The HMM for unsupervised POS induction is given by

$$P(\mathbf{x};\boldsymbol{\theta}) = \sum_{\mathbf{z}} p(\mathbf{z}_1) p(\mathbf{x}_1|\mathbf{z}_1) \prod_{n=2}^{N} p(\mathbf{z}_n|\mathbf{z}_{n-1}) p(\mathbf{x}_n|\mathbf{z}_n) \tag{8}$$

where $p(z)$ is *initial* probability, $p(z'|z)$ is *transition* probability, and $p(x|z)$ is *emission* probability. These parameters are required to be normalized: $\sum_z p(z) = 1$, $\forall z : \sum_{z'} p(z'|z) = 1$, and $\forall z : \sum_x p(x|z) = 1$.

Maximum Likelihood Estimation The partial derivatives of $\log P(\mathbf{x};\boldsymbol{\theta})$ with respect to model parameters are

$$\frac{\partial \log P(\mathbf{x};\boldsymbol{\theta})}{\partial p(z)} = \frac{1}{p(z)} \mathbb{E}_{\mathbf{z}|\mathbf{x};\boldsymbol{\theta}} \Big[\delta(\mathbf{z}_1, z) \Big] \tag{9}$$

$$\frac{\partial \log P(\mathbf{x};\boldsymbol{\theta})}{\partial p(z'|z)} = \frac{1}{p(z'|z)} \mathbb{E}_{\mathbf{z}|\mathbf{x};\boldsymbol{\theta}} \left[\sum_{n=2}^{N} \delta(\mathbf{z}_{n-1}, z) \delta(\mathbf{z}_n, z') \right] \tag{10}$$

$$\frac{\partial \log P(\mathbf{x};\boldsymbol{\theta})}{\partial p(x|z)} = \frac{1}{p(x|z)} \mathbb{E}_{\mathbf{z}|\mathbf{x};\boldsymbol{\theta}} \left[\sum_{n=1}^{N} \delta(\mathbf{x}_n, x) \delta(\mathbf{z}_n, z) \right] \tag{11}$$

Clearly, maximum likelihood estimators focus on calculating event expectations (e.g., the expected count of the event that the first tag $\mathbf{z}_1$ happens to be z). These expectations can be exactly and efficiently calculated using dynamic programming algorithms.

Maximum Reconstruction Estimation The reconstruction probability for HMMs can be written as

$$P(\mathbf{x}|\mathbf{x};\boldsymbol{\theta}) = \sum_{\mathbf{z}} P(\mathbf{z}|\mathbf{x};\boldsymbol{\theta})P(\mathbf{x}|\mathbf{z};\boldsymbol{\theta}) \tag{12}$$

Note that $P(\mathbf{x}|\mathbf{z};\boldsymbol{\theta}) = \prod_{n=1}^{N} p(\mathbf{x}_n|\mathbf{z}_n)$ is readily defined in HMMs. Since the posterior probability $P(\mathbf{z}|\mathbf{x};\boldsymbol{\theta})$ is not straightforward to calculate, we resort to the Bayes theorem instead:

$$P(\mathbf{z}|\mathbf{x};\boldsymbol{\theta}) = \frac{P(\mathbf{z};\boldsymbol{\theta})P(\mathbf{x}|\mathbf{z};\boldsymbol{\theta})}{P(\mathbf{x};\boldsymbol{\theta})} \tag{13}$$

where $P(\mathbf{z};\boldsymbol{\theta}) = p(\mathbf{z}_1)\prod_{n=2}^{N} p(\mathbf{z}_n|\mathbf{z}_{n-1})$.

Therefore, Eq. (12) can be re-written as

$$P(\mathbf{x}|\mathbf{x};\boldsymbol{\theta}) = \frac{\sum_{\mathbf{z}} P(\mathbf{z};\boldsymbol{\theta})P(\mathbf{x}|\mathbf{z};\boldsymbol{\theta})^2}{P(\mathbf{x};\boldsymbol{\theta})} \tag{14}$$

Note that all terms in the right-hand side of Eq. (14) are differentiable with respect to model parameters now. This is critical for directly learning data distribution parameters using the MRE criterion.

The partial derivatives of $\log P(\mathbf{x}|\mathbf{x};\boldsymbol{\theta})$ with respect to model parameters are

$$\begin{aligned}&\frac{\partial \log P(\mathbf{x}|\mathbf{x};\boldsymbol{\theta})}{\partial p(z)}\\ =\ & \frac{1}{p(z)}\Big(\mathbb{E}_Q\big[\delta(\mathbf{z}_1,z)\big] - \mathbb{E}_{\mathbf{z}|\mathbf{x};\boldsymbol{\theta}}\big[\delta(\mathbf{z}_1,z)\big]\Big)\end{aligned} \tag{15}$$

$$\begin{aligned}&\frac{\partial \log P(\mathbf{x}|\mathbf{x};\boldsymbol{\theta})}{\partial p(z'|z)}\\ =\ & \frac{1}{p(z'|z)}\Bigg(\mathbb{E}_Q\Big[\sum_{n=2}^{N}\delta(\mathbf{z}_{n-1},z)\delta(\mathbf{z}_n,z')\Big] -\\ & \qquad \mathbb{E}_{\mathbf{z}|\mathbf{x};\boldsymbol{\theta}}\Big[\sum_{n=2}^{N}\delta(\mathbf{z}_{n-1},z)\delta(\mathbf{z}_n,z')\Big]\Bigg)\end{aligned} \tag{16}$$

$$\begin{aligned}&\frac{\partial \log P(\mathbf{x}|\mathbf{x};\boldsymbol{\theta})}{\partial p(x|z)}\\ =\ & \frac{1}{p(x|z)}\Bigg(\mathbb{E}_Q\Big[\sum_{n=1}^{N}2\delta(\mathbf{x}_n,x)\delta(\mathbf{z}_n,z)\Big] -\\ & \qquad \mathbb{E}_{\mathbf{z}|\mathbf{x};\boldsymbol{\theta}}\Big[\sum_{n=1}^{N}\delta(\mathbf{x}_n,x)\delta(\mathbf{z}_n,z)\Big]\Bigg)\end{aligned} \tag{17}$$

where the distribution Q is defined as

$$Q(\mathbf{x},\mathbf{z};\boldsymbol{\theta}) = \frac{P(\mathbf{z};\boldsymbol{\theta})P(\mathbf{x}|\mathbf{z};\boldsymbol{\theta})^2}{\sum_{\mathbf{z}'}P(\mathbf{z}';\boldsymbol{\theta})P(\mathbf{x}|\mathbf{z}';\boldsymbol{\theta})^2} \tag{18}$$

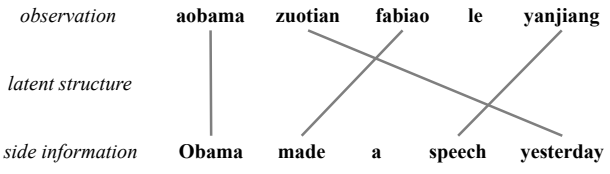

Figure 2: Word alignment. Given a (romanized) Chinese sentence and an English sentence, the task is to identify the latent correspondence between Chinese and English words.

While MLE calculates the expected count of an event with respect to the posterior distribution, MRE calculates the difference between the expected count with respect to the Q distribution and that with respect to the posterior distribution. Since the difference can be negative, we use the exponentiated gradient algorithm (Kivinen and Warmuth 1997) with adaptive learning rate (Bagos, Liakopoulos, and Hamodrakas 2004) for estimating model parameters. The expectations with respect to Q can also be exactly and efficiently calculated using dynamic programming algorithms (see Appendix A).

IBM Translation Models for Unsupervised Word Alignment

IBM translation models (Brown et al. 1993) provide a principled mechanism for translating between natural languages. The latent structure in IBM models is *word alignment*, which identifies the correspondence between words in two languages. For example, Figure 2 shows a (romanized) Chinese sentence "aobama zuotian fabiao le yanjiang" and an English sentence "Obama made a speech yesterday". IBM models can be used to compute the latent correspondence between Chinese and English words. Note that we follow Ammar, Dyer, and Smith (2014) to treat the Chinese sentence as observation and the English sentence as side information.

More formally, let $\mathbf{x} = \mathbf{x}_1,\ldots,\mathbf{x}_n,\ldots,\mathbf{x}_N$ be a foreign language sentence with N words and $\mathbf{y} = \mathbf{y}_1,\ldots,\mathbf{y}_m,\ldots,\mathbf{y}_M$ be an English sentence with M words. As Brown et al. (1993) restrict that each foreign word is aligned to exactly one English word, an alignment is defined as $\mathbf{z} = \mathbf{z}_1,\ldots,\mathbf{z}_n,\ldots,\mathbf{z}_N$, where $\mathbf{z}_n \in \{0,1,\ldots,M\}$. For example, $\mathbf{z}_n = m$ denotes that $\mathbf{x}_n$ connects to $\mathbf{y}_m$. Note that $\mathbf{y}_0$ is an empty cept. For example, in Figure 2, the Chinese word "le" is aligned to an empty cept (i.e., unaligned graphically).

In this work, we focus on IBM Model 2:

$$P(\mathbf{x}|\mathbf{y};\boldsymbol{\theta}) = \sum_{\mathbf{z}} P(\mathbf{x},\mathbf{z}|\mathbf{y};\boldsymbol{\theta}) \tag{19}$$

$$= \sum_{\mathbf{z}} \epsilon \prod_{n=1}^{N} p(\mathbf{z}_n|n,M,N)p(\mathbf{x}_n|\mathbf{y}_{\mathbf{z}_n}) \tag{20}$$

where $p(x|y)$ is *translation* probability, $p(m'|n',M',N')$ is *alignment* probability, and ϵ is a small fixed number

for the *length* sub-model. These parameters are required to be normalized: $\forall y : \sum_x p(x|y) = 1$ and $\forall n', M', N' : \sum_{m'} p(m'|n', M', N') = 1$.

Maximum Likelihood Estimation The partial derivatives of $\log P(\mathbf{x}|\mathbf{y})$ with respect to model parameters are

$$\frac{\partial \log P(\mathbf{x}|\mathbf{y};\boldsymbol{\theta})}{\partial p(x|y)} = \frac{1}{p(x|y)} \mathbb{E}_{\mathbf{z}|\mathbf{x},\mathbf{y};\boldsymbol{\theta}} \left[\sum_{n=1}^{N} \delta(\mathbf{x}_n, x)\delta(\mathbf{y}_{\mathbf{z}_n}, y) \right] \quad (21)$$

$$\frac{\partial \log P(\mathbf{x}|\mathbf{y};\boldsymbol{\theta})}{\partial p(m'|n', M', N')} = \frac{\delta(M', M)\delta(N', N)}{p(m'|n', M', N')} \mathbb{E}_{\mathbf{z}|\mathbf{x},\mathbf{y};\boldsymbol{\theta}} \left[\sum_{n=1}^{N} \delta(\mathbf{z}_n, m')\delta(n, n') \right] \quad (22)$$

Please refer to (Brown et al. 1993) for details about calculating the expectations.

Maximum Reconstruction Estimation Although both $\mathbf{x}$ and $\mathbf{y}$ are observed, we are only interested in reconstructing $\mathbf{x}$ because of the goal of machine translation: translating $\mathbf{y}$ to $\mathbf{x}$. The reconstruction probability is defined as

$$P(\mathbf{x}|\mathbf{x},\mathbf{y};\boldsymbol{\theta}) = \sum_{\mathbf{z}} P(\mathbf{z}|\mathbf{x},\mathbf{y};\boldsymbol{\theta}) P(\mathbf{x}|\mathbf{z},\mathbf{y};\boldsymbol{\theta}) \quad (23)$$

$$= \frac{\sum_{\mathbf{z}} P(\mathbf{x},\mathbf{z}|\mathbf{y};\boldsymbol{\theta}) P(\mathbf{x}|\mathbf{z},\mathbf{y};\boldsymbol{\theta})}{P(\mathbf{x}|\mathbf{y};\boldsymbol{\theta})} \quad (24)$$

where the probability of re-generating $\mathbf{x}$ given $\mathbf{y}$ and $\mathbf{z}$ is given by

$$P(\mathbf{x}|\mathbf{z},\mathbf{y};\boldsymbol{\theta}) = \prod_{n=1}^{N} p(\mathbf{x}_n|\mathbf{y}_{\mathbf{z}_n}) \quad (25)$$

Note that all terms in Eq. (24) are differentiable with respect to model parameters.

The partial derivatives of $\log P(\mathbf{x}|\mathbf{x},\mathbf{y};\boldsymbol{\theta})$ with respect to model parameters are

$$\frac{\partial \log P(\mathbf{x}|\mathbf{x},\mathbf{y};\boldsymbol{\theta})}{\partial p(x|y)} = \frac{1}{p(x|y)} \left(\mathbb{E}_Q \Big[\sum_{n=1}^{N} 2\delta(\mathbf{x}_n, x)\delta(\mathbf{y}_{\mathbf{z}_n}, y) \Big] - \mathbb{E}_{\mathbf{z}|\mathbf{x},\mathbf{y};\boldsymbol{\theta}} \Big[\sum_{n=1}^{N} \delta(\mathbf{x}_n, x)\delta(\mathbf{y}_{\mathbf{z}_n}, y) \Big] \right) \quad (26)$$

$$\frac{\partial \log P(\mathbf{x}|\mathbf{x},\mathbf{y};\boldsymbol{\theta})}{\partial p(m'|n', M', N')} = \frac{\delta(M', M)\delta(N', N)}{p(m'|n', M', N')} \times \left(\mathbb{E}_Q \Big[\sum_{n=1}^{N} \delta(\mathbf{z}_n, m')\delta(n, n') \Big] - \mathbb{E}_{\mathbf{z}|\mathbf{x},\mathbf{y};\boldsymbol{\theta}} \Big[\sum_{n=1}^{N} \delta(\mathbf{z}_n, m')\delta(n, n') \Big] \right) \quad (27)$$

# state	MLE		MRE	
	accuracy	VI	accuracy	VI
10	0.4054	3.0575	0.3881	2.9322
20	0.4804	3.1119	0.5203	2.8879
30	0.5341	3.0835	0.5653	2.8199
40	0.5817	3.1780	0.6191	2.9255
50	0.6108	3.2087	0.6739	2.7522

Table 1: Comparison of MLE and MRE on HMMs for unsupervised part-of-speech induction. The evaluation metrics are many-to-one accuracy (accuracy) and variation of information (VI).

where the distribution Q is defined as

$$Q(\mathbf{x},\mathbf{y},\mathbf{z};\boldsymbol{\theta}) = \frac{P(\mathbf{x},\mathbf{z}|\mathbf{y};\boldsymbol{\theta}) P(\mathbf{x}|\mathbf{z},\mathbf{y};\boldsymbol{\theta})}{\sum_{\mathbf{z}'} P(\mathbf{x},\mathbf{z}'|\mathbf{y};\boldsymbol{\theta}) P(\mathbf{x}|\mathbf{z}',\mathbf{y};\boldsymbol{\theta})} \quad (28)$$

Note that Eq. (28) is equivalent to

$$Q(\mathbf{x},\mathbf{y},\mathbf{z};\boldsymbol{\theta}) = \frac{P(\mathbf{z}|\mathbf{y};\boldsymbol{\theta}) P(\mathbf{x}|\mathbf{z},\mathbf{y};\boldsymbol{\theta})^2}{\sum_{\mathbf{z}'} P(\mathbf{z}'|\mathbf{y};\boldsymbol{\theta}) P(\mathbf{x}|\mathbf{z}',\mathbf{y};\boldsymbol{\theta})^2} \quad (29)$$

The expectations with respect to Q can also be exactly and efficiently calculated (see Appendix B).

Experiments

We evaluated our approach on two unsupervised NLP tasks: part-of-speech induction and word alignment.

Evaluation on Part-of-Speech Induction

Setting We split the English Penn Treebank into two parts: 46K sentences for training and test and 1K sentences for optimizing hyper-parameters of the exponentiated gradient (EG) algorithm with adaptive learning rate. Each word is manually labeled with a gold-standard part-of-speech tag. We used two evaluation metrics: *many-to-1 accuracy* (Johnson 2007) and *variation of information* (VI) (Beal 2003). The EM algorithm for maximum likelihood estimation runs for 100 iterations and the EG algorithm with adaptive learning rate runs for 50 iterations with initialization of a basic HMM (Ammar, Dyer, and Smith 2014). The number of hidden states in HMMs is set to 50, which is close to the size of the POS tag set.

Comparison with MLE Table 1 shows the comparison of MLE and MRE. With the increase of the number of hidden states, the expressiveness of HMMs generally improves accordingly. We find that MRE outperforms MLE for 50-state HMMs in terms of both *many-to-one accuracy* and *VI*, suggesting that our approach is capable of guiding the HMMs to use latent structures to find intended correlations in the data. The differences are statistically significant ($p < 0.01$). On average, the reconstruction probability of training examples using model parameters learned by MLE (i.e., $P(\mathbf{x}|\mathbf{x};\hat{\boldsymbol{\theta}}_{\mathrm{MLE}})$) is e^{-105}. In contrast, the average reconstruction probability by MRE (i.e., $P(\mathbf{x}|\mathbf{x};\hat{\boldsymbol{\theta}}_{\mathrm{MRE}})$) is e^{-84}.

Table 2 gives the results on training corpora with various sizes. Generally, the accuracy improves with the increase of

# sent.	MLE		MRE	
	accuracy	VI	accuracy	VI
10,000	0.5087	3.3471	0.5825	2.9018
20,000	0.5390	3.2387	0.5874	2.9217
30,000	0.5556	3.0764	0.6000	2.7904
40,000	0.5800	3.0117	0.6112	2.7403

Table 2: Effect of training corpus size.

MLE		MRE	
,	0.2077	said	0.4632
said	0.1514	says	0.0773
is	0.0371	reported	0.0326
says	0.0312	officials	0.0198
say	0.0307	announced	0.0195
:	0.0237	unit	0.0158
's	0.0203	noted	0.0119
think	0.0169	gained	0.0106
added	0.0129	told	0.0102
was	0.0129	court	0.0101

Table 3: Example emission probabilities for the POS tag "VBD" (verb past tense).

# state	CRF Autoencoders		MRE	
	accuracy	VI	accuracy	VI
10	0.4059	2.7145	0.3881	2.9322
20	0.4657	2.7462	0.5203	2.8879
30	0.5479	2.9585	0.5653	2.8199
40	0.5377	3.1048	0.6191	2.9255
50	0.5662	2.8450	0.6739	2.7522

Table 4: Comparison between CRF Autoencoders and MRE on unsupervised part-of-speech induction.

training corpus size for both MLE and MRE. The case for VI is similar. We find that our approach outperforms MLE consistently.

Table 3 shows example emission probabilities (e.g., $p(x|z)$) for the POS tag "VBD" (verb past tense). We follow Johnson (2007) to deterministically map hidden states to POS tags based on co-occurrence. As shown in Table 3, we find that MLE is prone to learn common but irrelevant correlations in the data (e.g., frequent words such as ",", ":", and "is"). In contrast, MRE is capable of identifying "said", "reported", "announced", "noted", "gained", and "told" correctly, suggesting that MRE enables HMMs to better discover intended correlations in the data.

Comparison with CRF Autoencoders We also compare our approach with CRF Autoencoders (Ammar, Dyer, and Smith 2014), which also builds on an encoding-reconstruction framework but allows for incorporating features. A surprising finding is that CRF Autoencoders achieves the highest accuracy with 50 states but obtains the lowest VI with 10 states. Our approach achieves the best accuracy and VI both with 50 states. While our approach slightly lags behind CRF Autoencoders in terms of VI, the improvements in terms of accuracy are statistically significant ($p < 0.01$).

criterion	model	C $\rightarrow$ E	E $\rightarrow$ C
MLE	Model 1	43.07	45.89
	Model 2	40.28	42.38
MRE	Model 1	41.90	45.39
	Model 2	38.33	41.73

Table 5: Comparison between MLE and MRE on IBM translation models for unsupervised word alignment. The evaluation metric is alignment error rate (AER).

MLE		MRE	
article	0.4932	article	0.5428
the	0.1924	articles	0.0995
says	0.0586	says	0.0624
points	0.0293	published	0.0497
an	0.0263	points	0.0349

Table 6: Example translation probabilities of the Chinese word "wenzhang".

Evaluation on Word Alignment

Setting We used the FBIS corpus as the training corpus, which contains 240K Chinese-English parallel sentences with 6.9M Chinese words and 8.9M English words. We used the TsinghuaAligner development and test sets (Liu and Sun 2015), which both contain 450 sentence pairs with gold-standard annotations. The evaluation metric is *alignment error rate* (AER) (Och and Ney 2003). Both MLE and MRE use the following training scheme: 5 iterations for IBM Model 1 and 5 iterations for IBM Model 2. As IBM Model 1 is a simplified version of IBM Model 2, the parameters of Model 1 at iteration 5 are used to initialize Model 2. We distinguish between two translation directions: Chinese-to-English (C $\rightarrow$ E) and English-to-Chinese (E $\rightarrow$ C).

Comparison with MLE Table 5 shows the comparison between MLE and MRE. We find that MRE outperforms MLE for both translation directions. All the differences are statistically significant ($p < 0.01$).

Table 6 shows example translation probabilities (i.e., $p(x|y)$) of the Chinese word "wenzhang" (i.e., "article"). We find that MLE tends to identify frequent words such as "the" and "an" as candidate translations while MRE finds more relevant candidate translations. This finding further confirms that MRE is more robust to common but irrelevant correlations.

Comparison with CRF Autoencoders On the same dataset, CRF Autoencoders achieve much lower AERs: 32.54 for C $\rightarrow$ E and 29.81 for E $\rightarrow$ C, respectively. The reason is that CRF Autoencoders are a discriminative latent-variable model capable of including more expressive IBM Model 4 as features. In contrast, our approach focuses on providing better training criterion for generative latent-

variable models such as IBM Model 2. [1] The training time for CRF Autoencoders is about 15 days while it only takes our approach 4 hours. Our generative models can also serve as central features in CRF autoencoders.

Conclusion

We have presented maximum reconstruction estimation for training generative latent-variable models such as hidden Markov models and IBM translation models. In the future, we plan to apply our approach to more generative latent-variable models such as probabilistic context-free grammars and explore the possibility of developing new training algorithms that minimize reconstruction errors.

Calculating Expectations for MRE Training of Hidden Markov Models

The expectations with respect to Q can also be exactly and efficiently calculated using dynamic programming algorithms. For example, consider the following expectation:

$$\mathbb{E}_Q\left[\sum_{n=2}^{N}\delta(\mathbf{z}_{n-1},z)\delta(\mathbf{z}_n,z')\right]$$

$$=\frac{\sum_{\mathbf{z}}P(\mathbf{z};\boldsymbol{\theta})P(\mathbf{x}|\mathbf{z};\boldsymbol{\theta})^2\sum_{n=2}^{N}\delta(\mathbf{z}_{n-1},z)\delta(\mathbf{z}_n,z')}{\sum_{\mathbf{z}}P(\mathbf{z};\boldsymbol{\theta})P(\mathbf{x}|\mathbf{z};\boldsymbol{\theta})^2} \quad (30)$$

$$=\frac{\mathbb{E}_{\mathbf{z};\boldsymbol{\theta}}\left[P(\mathbf{x}|\mathbf{z};\boldsymbol{\theta})^2\sum_{n=2}^{N}\delta(\mathbf{z}_{n-1},z)\delta(\mathbf{z}_n,z')\right]}{\mathbb{E}_{\mathbf{z};\boldsymbol{\theta}}\left[P(\mathbf{x}|\mathbf{z};\boldsymbol{\theta})^2\right]} \quad (31)$$

We re-define the *forward probability* $\alpha_n(z)$ as

$$\alpha_n(z) = \begin{cases} p(z)p(\mathbf{x}_1|z)^2 & \text{if } n=1 \\ \sum_{z'}\alpha_{n-1}(z')p(z|z')p(\mathbf{x}_n|z)^2 & \text{otherwise} \end{cases} \quad (32)$$

and the *backward probability* $\beta_n(z)$ as

$$\beta_n(z) = \begin{cases} 1 & \text{if } n=N \\ \sum_{z'}p(z|z')p(\mathbf{x}_{n+1}|z')^2\beta_{n+1}(z') & \text{otherwise} \end{cases} \quad (33)$$

Therefore, the expectation in the denominator of Eq. (31) can be re-written as

$$\mathbb{E}_{\mathbf{z};\boldsymbol{\theta}}\left[P(\mathbf{x}|\mathbf{z};\boldsymbol{\theta})^2\right]$$

$$=\sum_{\mathbf{z}}p(\mathbf{z}_1)\prod_{n=2}^{N}p(\mathbf{z}_n|\mathbf{z}_{n-1})\left(\prod_{n=1}^{N}p(\mathbf{x}_n|\mathbf{z}_n)\right)^2 \quad (34)$$

$$=\sum_{z}\alpha_N(z) \quad (35)$$

and the expectation in the numerator of Eq. (31) can be re-written as

$$\mathbb{E}_{\mathbf{z};\boldsymbol{\theta}}\left[P(\mathbf{x}|\mathbf{z};\boldsymbol{\theta})^2\sum_{n=2}^{N}\delta(\mathbf{z}_{n-1},z)\delta(\mathbf{z}_n,z')\right]$$

$$=\sum_{\mathbf{z}}p(\mathbf{z}_1)\prod_{n=2}^{N}p(\mathbf{z}_n|\mathbf{z}_{n-1})\left(\prod_{n=1}^{N}p(\mathbf{x}_n|\mathbf{y}_n)\right)^2\times \sum_{n=2}^{N}\delta(\mathbf{z}_{n-1},z)\delta(\mathbf{z}_n,z') \quad (36)$$

$$=\sum_{n'=2}^{N}\sum_{\mathbf{z}}p(\mathbf{z}_1)\prod_{n=2}^{N}p(\mathbf{z}_n|\mathbf{z}_{n-1})\left(\prod_{n=1}^{N}p(\mathbf{x}_n|\mathbf{z}_n)\right)^2\times \delta(\mathbf{z}_{n'-1},z)\delta(\mathbf{z}_{n'},z') \quad (37)$$

$$=\sum_{n=2}^{N}\alpha_{n-1}(z)p(z'|z)p(\mathbf{x}_n|z')\beta_n(z') \quad (38)$$

Similarly, other expectations with respect to the Q distribution can also be calculated using forward and backward probabilities.

Calculating Expectations for MRE Training of IBM Model 2

The expectations with respect to Q can also be exactly and efficiently calculated. Consider the following expectation:

$$\mathbb{E}_Q\left[\sum_{n=1}^{N}\delta(\mathbf{x}_n,x)\delta(\mathbf{y}_{\mathbf{z}_n},y)\right]$$

$$=\frac{\sum_{\mathbf{z}}P(\mathbf{x},\mathbf{z}|\mathbf{y};\boldsymbol{\theta})P(\mathbf{x}|\mathbf{z},\mathbf{y};\boldsymbol{\theta})\sum_{n=1}^{N}\delta(\mathbf{x}_n,x)\delta(\mathbf{y}_{\mathbf{z}_n},y)}{\sum_{\mathbf{z}}P(\mathbf{x},\mathbf{z}|\mathbf{y};\boldsymbol{\theta})P(\mathbf{x}|\mathbf{z},\mathbf{y};\boldsymbol{\theta})} \quad (39)$$

The denominator of Eq. (39) can be calculated as

$$\sum_{\mathbf{z}}P(\mathbf{x},\mathbf{z}|\mathbf{y};\boldsymbol{\theta})P(\mathbf{x}|\mathbf{z},\mathbf{y};\boldsymbol{\theta})$$

$$=\sum_{\mathbf{z}}\epsilon\prod_{n=1}^{N}p(\mathbf{z}_n|n,M,N)p(\mathbf{x}_n|\mathbf{y}_{\mathbf{z}_n})^2 \quad (40)$$

$$=\epsilon\prod_{n=1}^{N}\sum_{z=0}^{M}p(z|n,M,N)p(\mathbf{x}_n|\mathbf{y}_z)^2 \quad (41)$$

The numerator of Eq. (39) can be efficiently calculated as

$$\sum_{\mathbf{z}}P(\mathbf{x},\mathbf{z}|\mathbf{y};\boldsymbol{\theta})P(\mathbf{x}|\mathbf{z},\mathbf{y};\boldsymbol{\theta})\sum_{n=1}^{N}\delta(\mathbf{x}_n,x)\delta(\mathbf{y}_{\mathbf{z}_n},y)$$

$$=\sum_{\mathbf{z}}\epsilon\prod_{n=1}^{N}p(\mathbf{z}_n|n,M,N)p(\mathbf{x}_n|\mathbf{y}_{\mathbf{z}_n})^2\times \sum_{n=1}^{N}\delta(\mathbf{x}_n,x)\delta(\mathbf{y}_{\mathbf{z}_n},y) \quad (42)$$

$$=\sum_{n'=1}^{N}\delta(\mathbf{x}_{n'},x)\epsilon\prod_{n=1}^{N}\sum_{z=0}^{M}p(z|n,M,N)p(\mathbf{x}_n|\mathbf{y}_z)^2\times \Big(\delta(n,n')\delta(\mathbf{y}_z,y)+1-\delta(n,n')\Big) \quad (43)$$

Other expectations with respect to the Q distribution can also be calculated similarly.

[1] Note that it is possible to apply our approach to IBM Model 4. We leave this for future work.

Acknowledgements

This research is supported by the 863 Program (2015AA015407), the National Natural Science Foundation of China (No. 61522204, 61361136003, 61532001), 1000 Talent Plan grant, Tsinghua Initiative Research Program grants 20151080475, a Google Faculty Research Award and Fund from Online Education Research Center, Ministry of Education (No. 2016ZD102). We sincerely thank the viewers for their valuable suggestions.

References

Alain, G.; Bengio, Y.; Yao, L.; Yosinski, J.; Thibodeau-Laufer, E.; Zhang, S.; and Vincent, P. 2015. Gsns: Generative stochastic networks. arXiv: 1503.05571.

Ammar, W.; Dyer, C.; and Smith, N. 2014. Conditional random field autoencoders for unsupervised structred prediction. In *Proceedings of NIPS 2014*.

Bagos, P.; Liakopoulos, T.; and Hamodrakas, S. 2004. Faster gradient descent training of hidden markov models, using individual learning rate adaptation. In *Grammatical Inference: Algorithms and Applications*. Springer.

Beal, M. J. 2003. *Variational algorithms for approximate Bayesian inference*. University of London.

Bengio, Y. 2009. Learning deep architectures for ai. *Foundations and Trends in Machin Learning*.

Brown, P.; Della Pietra, S.; Della Pietra, V.; and Mercer, R. 1993. The mathematics of statistical machine translation: Parameter estimation. *Computational Linguistics*.

Ganchev, K.; Graça, J. a.; Gillenwater, J.; and Taskar, B. 2010. Posterior regularization for structured latent variable models. *Journal of Machine Learning Research*.

Hinton, G.; Osindero, S.; and Teh, Y. 2006. Reducing the dimensionality of data with neural networks. *Science*.

Johnson, M. 2007. Why doesn't em find good hmm pos taggers. In *Proceedings of EMNLP 2007*.

Kivinen, J., and Warmuth, M. 1997. Exponentiated gradient versus gradient descent for linear predictors. *Information and Computation*.

Klein, D., and Manning, C. 2004. Corpus-based induction of syntactic structure: Models of dependency and consistency. In *Proceedings of ACL 2004*.

Liu, Y., and Sun, M. 2015. Contrastive unsupervised word alignment with non-local features. In *Proceedings of AAAI 2015*.

Merialdo, B. 1994. Tagging english text with a probabilistic model. *Computational Linguistics*.

Och, F., and Ney, H. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*.

Poon, H., and Domingos, P. 2009. Unsupervised semantic parsing. In *Proceedings of EMNLP 2009*.

Smith, N., and Eisner, J. 2005. Contrastive estimation: Training log-linear models on unlabeled data. In *Proceedings of ACL 2005*.

Socher, R.; Huang, E.; Pennington, J.; Ng, A.; and Manning, C. 2011. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In *Proceedings of NIPS 2011*.

Vincent, P.; Larochelle, H.; Bengio, Y.; and Manzagol, P. 2008. Extracting and composing robust features with denoising autoencoders. In *Proceedings of ICML 2008*.

Vincent, P.; H., L.; Lajoie, I.; Bengio, Y.; and Manzagol, P. 2010. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of Machine Learning Research*.

Vogel, S.; Ney, H.; and Tillmann, C. 1996. Hmm-based word alignment in statistical translation. In *Proceedings of COLING 1996*.