Deep Probabilistic Models - Tools

Deep Learning 2 – 2023

Wilker Aziz
w.aziz@uva.nl


University of Amsterdam
Institute for Logic, Language and Computation

Goals for this session

1. Parameterise complex distributions using neural networks

Outline

Prescribing distributions

We will now discuss various ways to prescribe distributions using deep learning. For each technique, we will keep an eye on two things:

- our ability to assess the probability mass/density of a given outcome
- our ability to sample outcomes from the corresponding distribution

We begin with the univariate case and then discuss the multivariate case.

Direct (aka 'locally normalised')

Predict the parameter of a known pdf (or pmf).

Temperature

Input: an image $\mathbf{x} \in \mathbb{R}^{H \times W \times C}$ of a car engine

Output: a distribution $\mathcal{N}(\mu(\mathbf{x}), \sigma(\mathbf{x})^2)$ over temperature values in Celsius where the Gaussian location and scale are predicted given $\mathbf{x}$ as follows

$$\begin{aligned} \mathbf{h} &= \text{encode}_D(\mathbf{x}; \theta_{\text{enc}}) \\ \mu(\mathbf{x}) &= \text{linear}_1(\mathbf{h}; \theta_{\text{loc}}) \\ \sigma(\mathbf{x}) &= \text{softplus}(\text{linear}_1(\mathbf{h}; \theta_{\text{scale}})) \end{aligned} \tag{1}$$

Direct (aka 'locally normalised')

Predict the parameter of a known pdf (or pmf).

- density/mass assessment is typically feasible (by design)
- sampling is typically possible (via inverse cdf method, or some specialised algorithm)

Heard count

Input: an image $\mathbf{x} \in \mathbb{R}^{H \times W \times C}$ of a field

Output: a distribution $\mathrm{Poisson}(\lambda(\mathbf{x}))$ over the number of cows in the field, where the Poisson rate is predicted given $\mathbf{x}$ as follows

$$\begin{aligned} \mathbf{h} &= \mathrm{encode}_D(\mathbf{x}; \theta_{\mathsf{enc}}) \\ \lambda(\mathbf{x}) &= \mathrm{softplus}(\mathrm{linear}_1(\mathbf{h}; \theta_{\mathsf{rate}})) \end{aligned} \tag{2}$$

CDF

Parameterise a cdf F (maps from an outcome to the cumulative probability p)

- density assessment: requires differentiation (which can be automated!)
- sampling: easy if analytically invertible (which is rare)

Temperature

Let $F_0(y; \phi) = \exp(-\exp(\phi - y))$ be the CDF of a Gumbel distribution with location ϕ.

Input: an image $\mathbf{x} \in \mathbb{R}^{H \times W \times C}$ of a car engine

Output: a convex combination of K simple CDFs

$$F(y) = \sum_{k=1}^{K} w_k F_0(y; \phi_k) \tag{3}$$

where $\mathbf{w} \in \Delta_{K-1}$ and $\boldsymbol{\phi} \in \mathbb{R}^K$ are predicted given $\mathbf{x}$ as follows

$$\mathbf{h} = \text{encode}_D(\mathbf{x}; \theta_{\text{enc}}) \tag{4}$$
$$\boldsymbol{\phi} = \text{linear}_K(\mathbf{h}; \theta_{\text{locs}}) \tag{5}$$
$$\mathbf{w} = \text{softmax}(\text{linear}_K(\mathbf{h}; \theta_{\text{mix}})) \tag{6}$$
$$\tag{7}$$

Sampler - inverse cdf

Parameterise a quantile function Q (maps from a probability value p to the outome)

- density assessment: difficult
- sampling: easy by design
  for a pair (p, y) such that $y = Q(p; \theta)$, the inverse function theorem enables density assessment

Temperature
Let $Q_0(p; \phi) = \phi - \log(-\log(p))$ be the quantile function (inverse CDF) of a Gumbel distribution with location ϕ.

Input: an image $\mathbf{x} \in \mathbb{R}^{H \times W \times C}$ of a car engine

Output: a conical combination of K simple quantile functions

$$Q(p) = \sum_{k=1}^{K} w_k Q_0(p; \phi_k) \tag{8}$$

where $\mathbf{w} \in \mathbb{R}^K_{>0}$ and $\boldsymbol{\phi} \in \mathbb{R}^K$ are predicted given $\mathbf{x}$ as follows

$$\mathbf{h} = \text{encode}_D(\mathbf{x}; \theta_{\text{enc}}) \tag{9}$$
$$\boldsymbol{\phi} = \text{linear}_K(\mathbf{h}; \theta_{\text{locs}}) \tag{10}$$
$$\mathbf{w} = \text{softplus}(\text{linear}_K(\mathbf{h}; \theta_{\text{mix}})) \tag{11}$$
$$\tag{12}$$

Sampler - bijection

Parameterise a bijective transformation of a known and convenient base random variable

- Assessment and sampling can be made simple, in some cases only one of the two is simple;
- Parameterising one such model takes some skill (to achieve efficient computations)

See Normalising Flows in our Advanced Generative Models module.

Suppose access to an invertible function (a bijection): $x = h^{-1}(y)$, then

$$p_Y(y) = p_X(h^{-1}(y))|\det J_{h^{-1}}(y)| \tag{13}$$

Start from a known p_X, for example a Gaussian, and obtain a novel p_Y, more complex than a Gaussian.

It's possible to assess the density of some outcome y by mapping it to the corresponding $x = h^{-1}(y)$, assessing its density and the Jacobian.

It's possible to draw samples, by drawing from the simple p_X and then mapping to the corresponding $y = h(x)$.

Sampler - general case

Parameterise an arbitrary transformation of a base random variable

- Sampling: trivial by design (it costs a forward pass through an NN we choose)
- Assessment: intractable in general! For an outcome $y \in \mathbb{R}^O$, a base density $p_X(x)$ on $\mathbb{R}^I$

$$p_Y(y) = \int_{\mathcal{C}} p_X(x) \mathrm{d}x \tag{14}$$

$$\mathcal{C} = \{x : f(x; \theta) = \mathbf{y}\} \tag{15}$$

$\mathcal{C}$ is the set of all points in $\mathbb{R}^I$ that f maps to exactly y

See Generative Adversarial Networks (GANs) in our Advanced Generative Models module

Here f is some deep neural network with I inputs and O outputs (e.g., a feed forward neural network)

$$x \sim \mathcal{N}(0, I) \tag{16}$$

$$y = f(x; \theta) \tag{17}$$

Unnormalised

Parameterise a non-negative measure or an energy

- Assessment: depends on the complexity of computing a normalisation constant
  - possibly tractable for finite countable sample spaces, but intractable in general
  - the normalisation constant might diverge for infinite sample spaces (leading to an improper distribution)
- Sampling: generally intractable, but possible for certain (simple) classes of models (e.g., first-order CRF). Tractable options include MCMC and approximate MCMC (e.g., stochastic gradient Langevin dynamics).

See energy-based models (EBM) in our Advanced Generative Models module

Input: a piece of text x about some entity and a set $\{y_n\}_{n=1}^N$ of variable size containing (potentially relevant) Wikipedia pages retrieved by an algorithm

Output: a discrete distribution assigning probability proportional to $\exp(\eta_n)$ to entity y_n being about the entity mentioned in x where

$$\mathbf{t} = \text{encode}_D(x; \theta_{\text{enc}}) \tag{18}$$

$$\text{for } n \in [N] \tag{19}$$

$$\mathbf{h}_n = \text{encode}_D(y_n; \theta_{\text{enc}}) \tag{20}$$

$$\mathbf{s}_n = \text{relu}(\text{linear}_H([\mathbf{t}, \mathbf{h}_n]; \theta_{\text{hid}})) \tag{21}$$

$$\eta_n = \text{linear}_1(\mathbf{s}_n; \theta_{\text{out}}) \tag{22}$$

Multivariate

For the fixed-dimension case, we may have access to multivariate generalisations of known pdfs (e.g., MVN).

In general (fixed-dimension or not), we can exploit a *factorisation* with or without conditional independence assumptions.

Then, we predict the factors:

- direct, or cdf, or sampler/simulator
- unnormalised

Examples of factorisation: $y = (w_1, \ldots, w_N)$

- full conditional independence $p_{Y|X}(y|x) \overset{\text{ind.}}{=} \prod_{n=1}^{N} p_{W|X}(w_n|x)$
- Markov model $p_{Y|X}(y|x) \overset{\text{ind.}}{=} \prod_{n=1}^{N} p_{W|XH}(w_n|x, w_{n-k+1:n-1})$
- chain rule $p_{Y|X}(y|x) = \prod_{n=1}^{N} P_{W|XH}(w_n|x, w_{<n})$
  aka *autoregressive factorisation*
- first-order conditional random field
  $p_{Y|X}(y|x) \propto \prod_{n=1}^{N} \Phi(x, y_{n-1}, y_n)$ where $\Phi(x, y_{n-1}, y_n) > 0$

Remarks

There are various ways to prescribe distributions both univariate and multivariate.

Ideas based on predicting parameters for known pdfs and cdfs, but also ideas based on parameterising simulators/samplers.

For multivariate and structured data, factorisation of complex joint distributions is key.

There are various tradeoffs: is mass/density assessment tractable? can we sample? do we need backward passes? do we need to solve normalisation constants?

Next class

Another tool for prescribing joint distributions: augment a joint distribution over observed random variables with unobserved random variables and perform marginal inference.

Modelling unobserved data (the case of discrete latent variables and exact inference)

References I

Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

Bryan Eikema and Wilker Aziz. Sampling-based minimum bayes risk decoding for neural machine translation. *arXiv preprint arXiv:2108.04718*, 2021.

Yarin Gal and Zoubin Ghahramani. Bayesian convolutional neural networks with Bernoulli approximate variational inference. In *ICLR - workshop track*, 2016a.

References II

Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In Maria Florina Balcan and Kilian Q. Weinberger, editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1050–1059, New York, New York, USA, June 2016b. PMLR. URL http://proceedings.mlr.press/v48/gal16.html.

Yarin Gal and Zoubin Ghahramani. A theoretically grounded application of dropout in recurrent neural networks. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in neural information processing systems 29*, pages 1019–1027. Curran Associates, Inc., 2016c. URL http://papers.nips.cc/paper/6241-a-theoretically-grounded-application-of-dropout-in-rec pdf.

References III

Daphne Koller and Nir Friedman. *Probabilistic Graphical Models*. MIT Press, 2009.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951. Publisher: JSTOR.

Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. URL https://proceedings.neurips.cc/paper/2012/file/05311655a15b75fab86956663e1819cd-Paper.pdf.

References IV

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15 (56):1929–1958, 2014. URL http://jmlr.org/papers/v15/srivastava14a.html.