
TensorFlow Probability

TensorFlow Probability is a library for probabilistic reasoning and statistical analysis in TensorFlow. As part of the TensorFlow ecosystem, TensorFlow Probability provides integration of probabilistic methods with deep networks, gradient-based inference via automatic differentiation, and scalability to large datasets and models via hardware acceleration (e.g., GPUs) and distributed computation.

TFP also works as “Tensor-friendly Probability” in pure JAX!: `from tensorflow_probability.substrates import jax as tfp` – Learn more here.

Our probabilistic machine learning tools are structured as follows.

Layer 0: TensorFlow. Numerical operations. In particular, the `LinearOperator` class enables matrix-free implementations that can exploit special structure (diagonal, low-rank, etc.) for efficient computation. It is built and maintained by the TensorFlow Probability team and is now part of `tf.linalg` in core TF.

Layer 1: Statistical Building Blocks

- Distributions (`tfp.distributions`): A large collection of probability distributions and related statistics with batch and broadcasting semantics. See the Distributions Tutorial.
- Bijectors (`tfp.bijectors`): Reversible and composable transformations of random variables. Bijectors provide a rich class of transformed distributions, from classical examples like the log-normal distribution to sophisticated deep learning models such as masked autoregressive flows.

Layer 2: Model Building

- Joint Distributions (e.g., `tfp.distributions.JointDistributionSequential`): Joint distributions over one or more possibly-interdependent distributions. For an introduction to modeling with TFP’s `JointDistributions`, check out this colab
- Probabilistic Layers (`tfp.layers`): Neural network layers with uncertainty over the functions they represent, extending TensorFlow Layers.

Layer 3: Probabilistic Inference

- Markov chain Monte Carlo (`tfp.mcmc`): Algorithms for approximating integrals via sampling. Includes Hamiltonian Monte Carlo, random-walk Metropolis-Hastings, and the ability to build custom transition kernels.
- Variational Inference (`tfp.vi`): Algorithms for approximating integrals via optimization.
- Optimizers (`tfp.optimizer`): Stochastic optimization methods, extending TensorFlow Optimizers. Includes Stochastic Gradient Langevin Dynamics.
- Monte Carlo (`tfp.monte_carlo`): Tools for computing Monte Carlo expectations.

TensorFlow Probability is under active development. Interfaces may change at any time.

Examples

See [tensorflow_probability/examples/](#) for end-to-end examples. It includes tutorial notebooks such as:

- Linear Mixed Effects Models. A hierarchical linear model for sharing statistical strength across examples.
- Eight Schools. A hierarchical normal model for exchangeable treatment effects.
- Hierarchical Linear Models. Hierarchical linear models compared among TensorFlow Probability, R, and Stan.
- Bayesian Gaussian Mixture Models. Clustering with a probabilistic generative model.
- Probabilistic Principal Components Analysis. Dimensionality reduction with latent variables.
- Gaussian Copulas. Probability distributions for capturing dependence across random variables.
- TensorFlow Distributions: A Gentle Introduction. Introduction to TensorFlow Distributions.
- Understanding TensorFlow Distributions Shapes. How to distinguish between samples, batches, and events for arbitrarily shaped probabilistic computations.
- TensorFlow Probability Case Study: Covariance Estimation. A user's case study in applying TensorFlow Probability to estimate covariances.

It also includes example scripts such as:

Representation learning with a latent code and variational inference. * Vector-Quantized Autoencoder. Discrete representation learning with vector quantization. * Disentangled Sequential Variational Autoencoder Disentangled representation learning over sequences with variational inference. * Bayesian Neural Networks. Neural networks with uncertainty over their weights. * Bayesian Logistic Regression. Bayesian inference for binary classification.

Installation

For additional details on installing TensorFlow, guidance installing prerequisites, and (optionally) setting up virtual environments, see the TensorFlow installation guide.

Stable Builds

To install the latest stable version, run the following:

```
1 # Notes:
2
3 # - The `--upgrade` flag ensures you'll get the latest version.
4 # - The `--user` flag ensures the packages are installed to your user
    directory
5 #   rather than the system directory.
6 # - TensorFlow 2 packages require a pip >= 19.0
7 python -m pip install --upgrade --user pip
8 python -m pip install --upgrade --user tensorflow
    tensorflow_probability
```

For CPU-only usage (and a smaller install), install with `tensorflow-cpu`.

To use a pre-2.0 version of TensorFlow, run:

```
1 python -m pip install --upgrade --user "tensorflow<2" "
    tensorflow_probability<0.9"
```

Note: Since TensorFlow is *not* included as a dependency of the TensorFlow Probability package (in `setup.py`), you must explicitly install the TensorFlow package (`tensorflow` or `tensorflow-cpu`). This allows us to maintain one package instead of separate packages for CPU and GPU-enabled TensorFlow. See the TFP release notes for more details about dependencies between TensorFlow and TensorFlow Probability.

Nightly Builds

There are also nightly builds of TensorFlow Probability under the pip package `tfp-nightly`, which depends on one of `tf-nightly` or `tf-nightly-cpu`. Nightly builds include newer features, but may be less stable than the versioned releases. Both stable and nightly docs are available [here](#).

```
1 python -m pip install --upgrade --user tf-nightly tfp-nightly
```

Installing from Source

You can also install from source. This requires the Bazel build system. It is highly recommended that you install the nightly build of TensorFlow (`tf-nightly`) before trying to build TensorFlow Probability from source. The most recent version of Bazel that TFP currently supports is 6.4.0; support for 7.0.0+ is WIP.

```
1 # sudo apt-get install bazel git python-pip # Ubuntu; others, see
    above links.
2 python -m pip install --upgrade --user tf-nightly
3 git clone https://github.com/tensorflow/probability.git
```

```
4 cd probability
5 bazel build --copt=-03 --copt=-march=native :pip_pkg
6 PKGDIR=$(mktemp -d)
7 ./bazel-bin/pip_pkg $PKGDIR
8 python -m pip install --upgrade --user $PKGDIR/*.whl
```

Community

As part of TensorFlow, we're committed to fostering an open and welcoming environment.

- Stack Overflow: Ask or answer technical questions.
- GitHub: Report bugs or make feature requests.
- TensorFlow Blog: Stay up to date on content from the TensorFlow team and best articles from the community.
- Youtube Channel: Follow TensorFlow shows.
- tfprobability@tensorflow.org: Open mailing list for discussion and questions.

See the TensorFlow Community page for more details. Check out our latest publicity here:

- Coffee with a Googler: Probabilistic Machine Learning in TensorFlow
- Introducing TensorFlow Probability

Contributing

We're eager to collaborate with you! See [CONTRIBUTING.md](#) for a guide on how to contribute. This project adheres to TensorFlow's code of conduct. By participating, you are expected to uphold this code.

References

If you use TensorFlow Probability in a paper, please cite:

- *TensorFlow Distributions*. Joshua V. Dillon, Ian Langmore, Dustin Tran, Eugene Brevdo, Srinivas Vasudevan, Dave Moore, Brian Patton, Alex Alemi, Matt Hoffman, Rif A. Saurous. arXiv preprint arXiv:1711.10604, 2017.

(We're aware there's a lot more to TensorFlow Probability than Distributions, but the Distributions paper lays out our vision and is a fine thing to cite for now.)