Review-45

Neural Ordinary Differential Equations

Driven by advances in deep layered architectures, the work introduces a novel family of models which optimize over its parameters by solving a neural Ordinary Differential Equation (ODE). The method parameterizes the derivative of the hidden state which results in computation of a black-box equation solver capable of approximating continuous depth models. Neural ODEs result in constant memory and cost as a result of eliminating backpropogation updates through operations of the solver. Neural ODEs extend towards residual networks, continuous-time latent variable models, continuous normalizing flows and maximum-likelihood generative models.

Conventional ODE solutions made use of Euler's method which has advanced towards efficient and accurate solvers. The work employs adjoint sensitivity method for solving ODE gradients. The solver is treated as a black-box with the adjoint $a(t)$ being the gradient of loss $L(\theta)$ with respect to hidden state $z(t)$. The dynamics of adjoint $\frac{da(t)}{dt}$ are obtained using instantaneous chain rule. Another call to the ODE solver computes $\frac{\partial L}{\partial z(t)}$ by running backwards from $z(t)$ to $z(t-1)$. Combining these gradients results in the final update $\frac{dL}{d\theta}$. The reverse-mode automatic differentiation of Neural ODEs provisions continuous implicit layers within the network which are otherwise found absent in architectures. The framework of continuous Neural ODEs is extended towards residual networks for supervised learning, continuous normalizing flows for maximum likelihood training and generative latent-variable models for time-series prediction.

Neurl ODEs demonstrate comparable performance with ResNets and RK-Net while depicting lower memory cost and number of parameters. ODEs extend towards continuous normalizing flows by making use of the change of variables theorem and present improved density matching when compared to planar normalizing flows. Additionally, ODEs demonstrate applications in generating missing data using latent variable models. The method employs an RNN encoder which samples a latent trajectory and solves the ODE corresponding to its points. Following the solution, the solver extrapolates by sampling from the constructed posterior to generate data. Provision of continuous variables allows the solver to learn a finer mapping in latent space which results in accurate extrapolation in predictions. While the continuous nature of ODEs demonstrates success in comparison to its discrete counterparts, the method provides room for future improvement. The work throws light on minibatching and asks questions upon what might it be its potential role in the case of continuous implicit layers. Secondly, ODEs tradeoff speed for precision and require setting an appropriate error tolerance during training. Lastly, the work indicates that reconstrution of trajectories results in numerical errors in case of divergence. However, the work does not comment on how can the divergence be minimized. Instead, the method offers a workaround of checkpointing the variables and reconstructing them upon integration.

Neural ODEs provide two novel directions for future research. Firstly, the setting of continuous implicit layers can be extended towards high-dimensional datasets in order to observe their tolerance towards errors and approximation quality with large batch sizes. Secondly, ODEs could be extended to study their limitations during forward trajectories which would aid in answering the question of reconstructing forward trajectories.

Motivated by innovations in deep-layered design, the work introduced a family of continuous Neural ODE models. The framework allows one to optimize a neural network's parameters by solving an ODE using the adjoint sensitivty method. This provisions memory efficiency and faster computation

at the cost of precision during reverse-mode automatic differentiation. Neural ODEs present efficient training of supervised and maximum likelihood learning with an extension towards trajectory generation using latent-variable models.