Location–Independent Weather Forecasting

Andrew Pillsbury and Rebecca Leong
CS 74 Machine Learning

Introduction

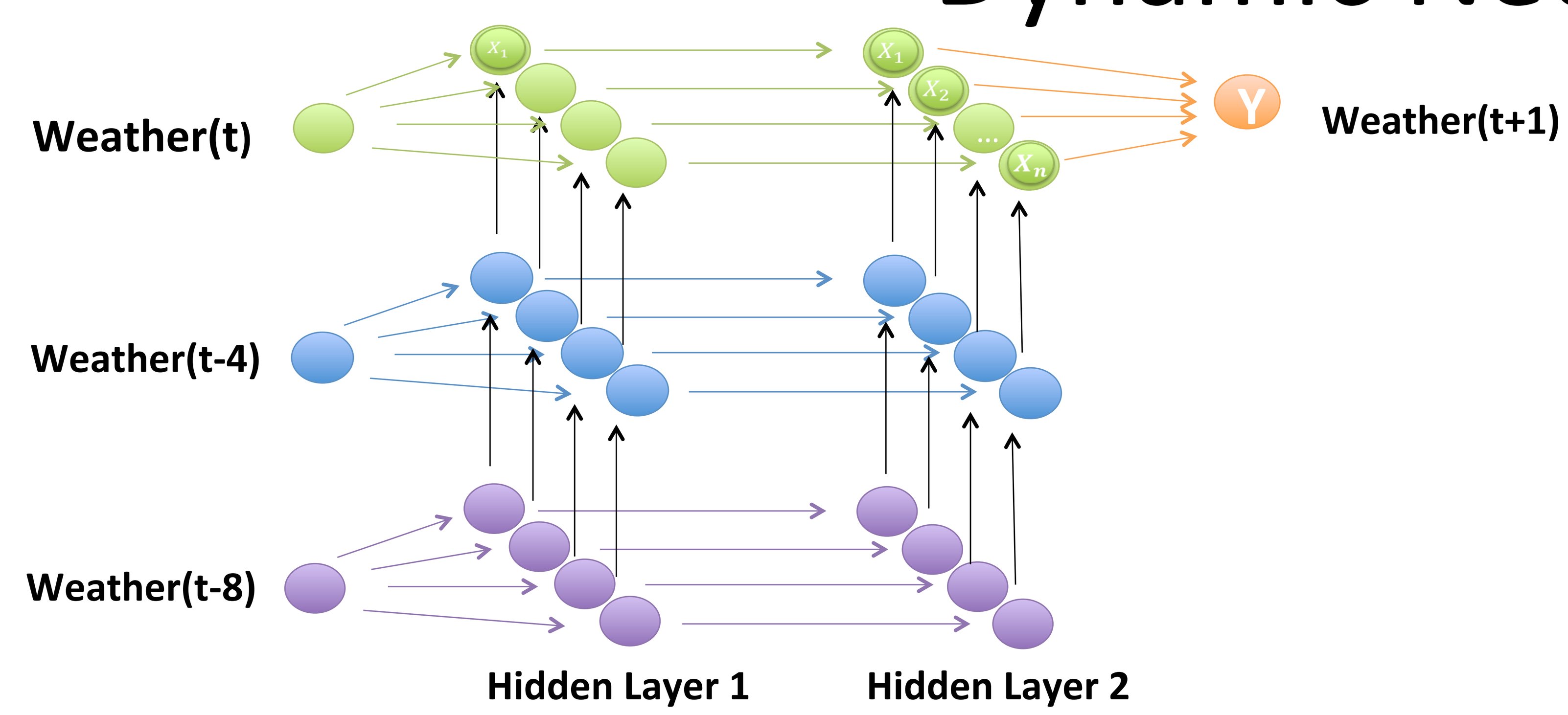
Weather forecasting has traditionally been treated as a well-studied physics-based phenomenon for a specific location. Even so, weather still exhibits data patterns that can potentially be utilized as a basis for future prediction. By gathering data from a variety of different locations in the continental United States, we are to train a dynamic neural network for location-neutral weather forecaster.

Objective

Input: The past three days of weather data samples recorded every four hours. Each sample includes readings of temperature, visibility, wind speed, wind direction, pressure, and dew point.

Output: The weather for the next day in 4 hour data samples.

Dynamic Neural Network



Feed Forward:

$$\mathbf{x}(n+1) = \mathbf{f}(\mathbf{W}^{in}\mathbf{u}(n+1) + \mathbf{W}\mathbf{x}(n)), \text{ where } f(x) = \tanh(x)$$

$$\mathbf{y}(n+1) = \mathbf{f}^{out}(\mathbf{W}^{out}(\mathbf{u}(n+1), \mathbf{x}(n+1)))$$

Monte Carlo Update Method:

For each weight matrix, update that matrix with a randomly generated matrix and check the error. If the error has gone down after the update, save the new matrix, otherwise discard it and try again.

Backpropagation Update Method:

Calculate deltas-

$$\delta_j(T) = (d_j(T) - y_j(T)) \left.\frac{\partial f(u)}{\partial u}\right|_{u=z_j(T)}$$

$$\delta_i(T) = \left[\sum_{j=1}^{L} \delta_j(T) w_{ji}^{out}\right] \left.\frac{\partial f(u)}{\partial u}\right|_{u=z_i(n)}$$

$$\delta_i(n) = \left[\sum_{j=1}^{N} \delta_j(n+1) w_{ji} + \sum_{j=1}^{L} \delta_j(n) w_{ji}^{out}\right] \left.\frac{\partial f(u)}{\partial u}\right|_{u=z_i(n)}$$

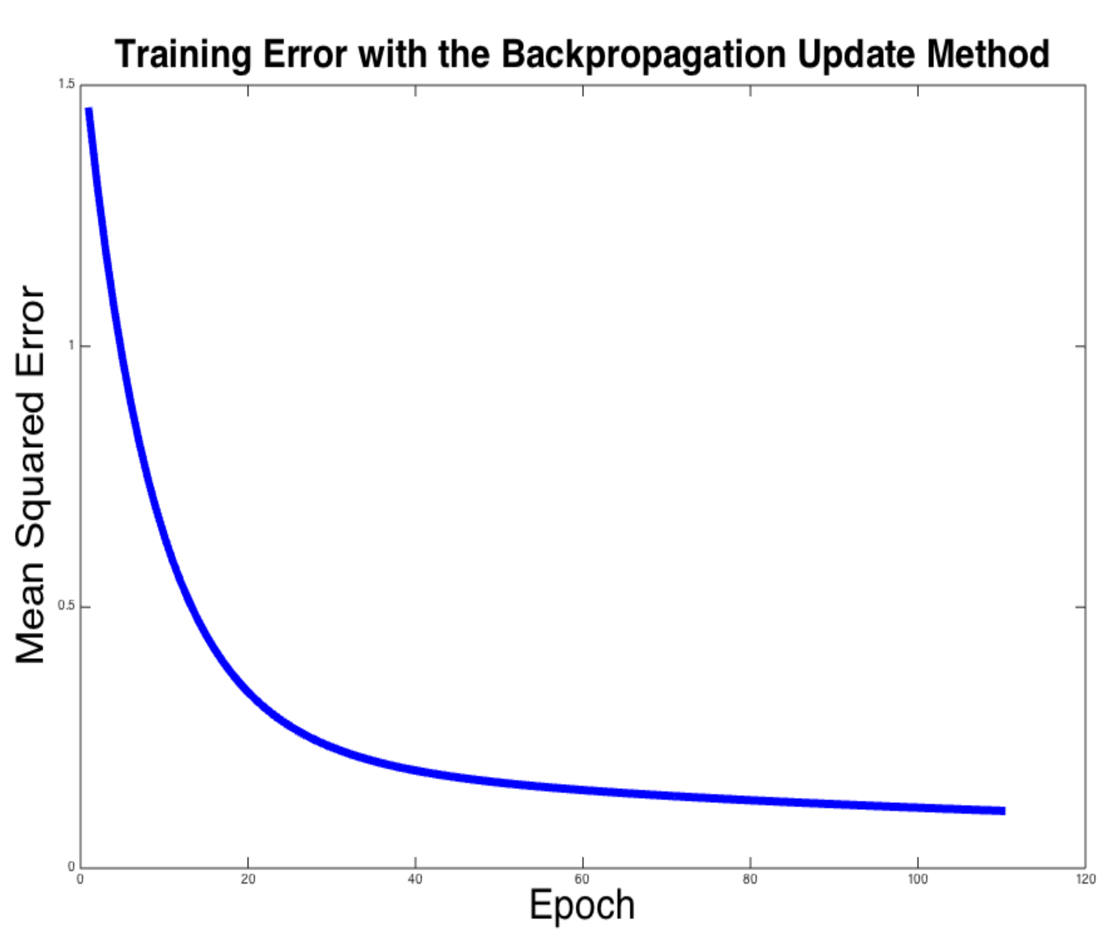
Update weights-

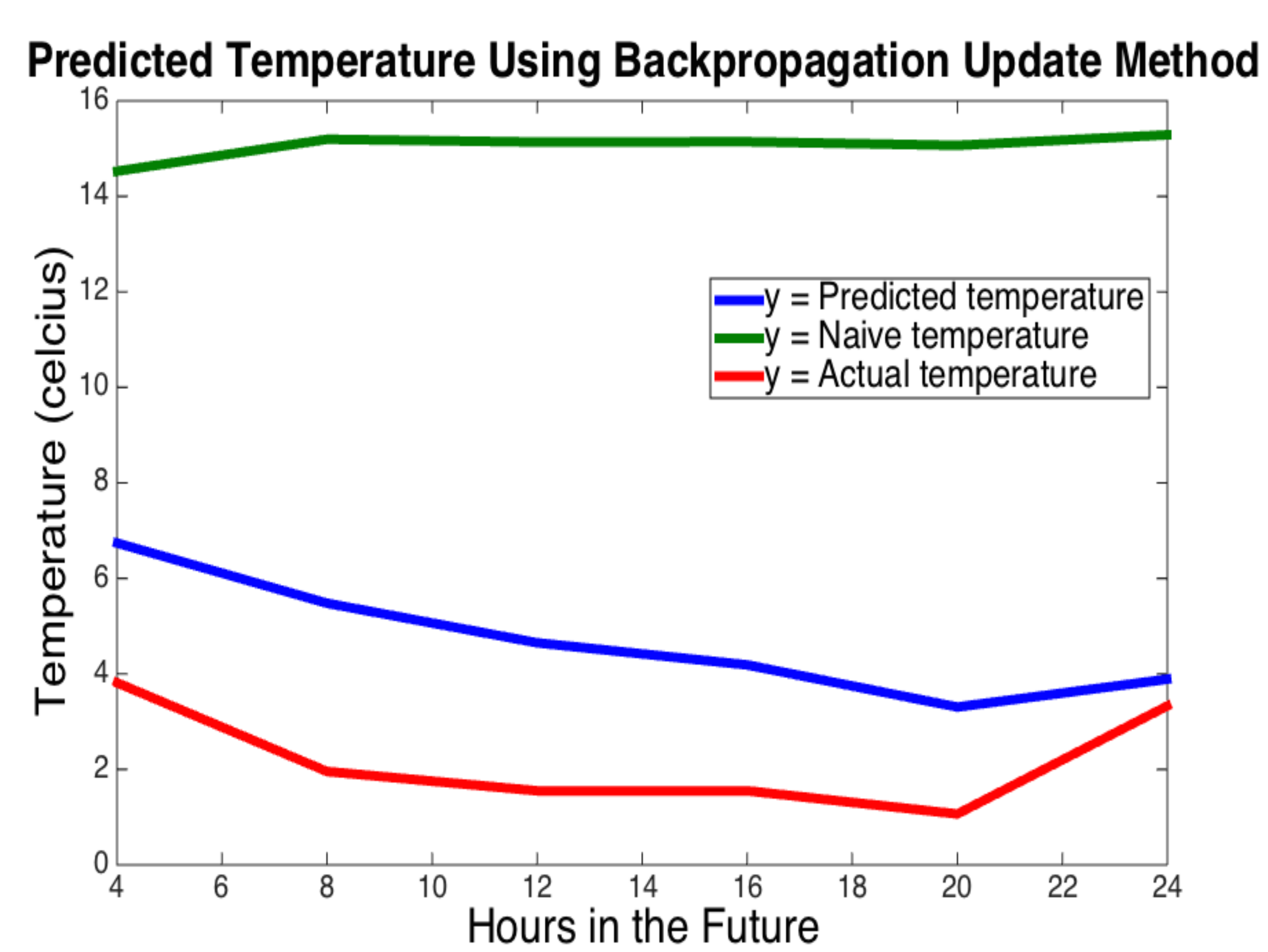
$$new\ w_{ij} = w_{ij} + \gamma \sum_{n=1}^{T} \delta_i(n) x_j(n-1) \quad [use\ x_j(n-1) = 0\ for\ n = 1]$$

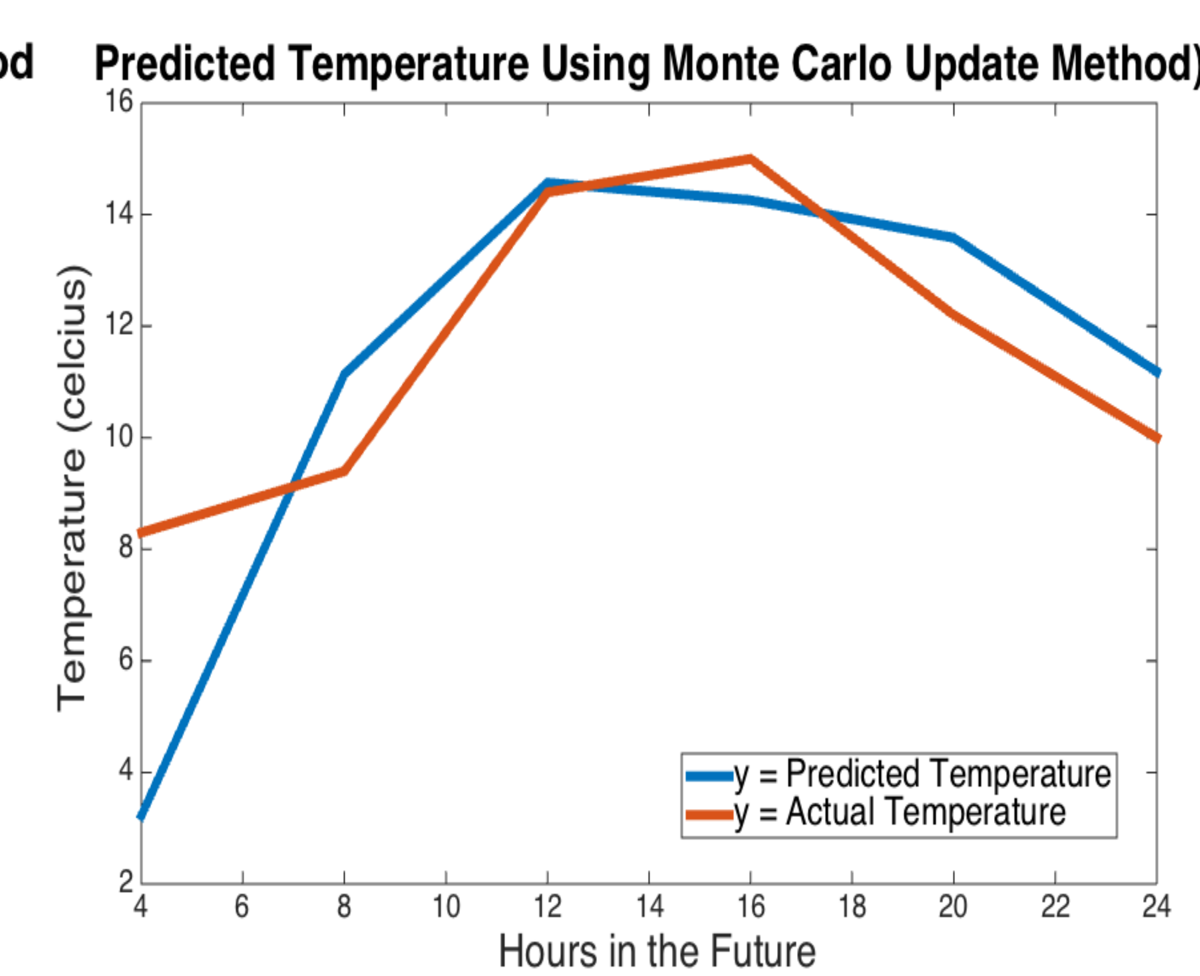
$$new\ w_{ij}^{in} = w_{ij}^{in} + \gamma \sum_{n=1}^{T} \delta_i(n) u_j(n)$$

$$new\ w_{ij}^{out} = w_{ij}^{out} + \gamma \times \begin{cases} \sum_{n=1}^{T} \delta_i(n) u_j(n), if\ \ j\ refers\ to\ input\ unit \\ \sum_{n=1}^{T} \delta_i(n) x_j(n), if\ \ j\ refers\ to\ hidden\ unit \end{cases}$$

Results







Future Work

- Run more tests to refine the three hyperparameters used in the network (number of neurons per layer, number of hidden layers, number of input samples).
- Expand the network to allow for an arbitrary number of hidden layers.
- Optimize for speed, potentially implementing a stochastic gradient descent.

Sources: H. Jaeger (2002, revised 2013): *Tutorial on training recurrent neural networks, covering BPPT, RTRL, EKF and the "echo state network" approach.* GMD Report 159, German National Research Center for Information Technology, 2002 (48 pp.)

Data Source: Quality Controlled Local Climatological Data from the National Climatic Data Center