Glaciology Research

Methods

Overview

Here we experiment with the application of neural networks to the task of predicting changes in mass balance for the Langjökull glacier during the summer and winter of 2015. We further integrate a Gaussian process on top of our network, incorporating spatio-temporal information to model prediction confidence. The goal is to utilize a limited number of historical mass balance observations along with observations for the season/year of interest to predict changes in mass balance across the entire glacier.

Neural Network

Our neural network jointly learns a temporally-invariant representation for each spatial region of the glacier (defined by longitude, latitude, and elevation), along with a time-dependent representation. This later component incorporates observations collected for the season/year of interest, along with for the previous year, consisting of two sequences of (longitude, latitude, elevation, mass balance) tuples. As the exact locations for each site aren't strictly fixed and vary in number by year, we utilize modern graph neural network methods to work with this data.

Our model is primarily constructed using base feed-forward neural network layers, defined as $h_i = \sigma(Norm(Wx_i^T + b))$, where x_i is the input, σ represents the activation function, and

Norm represents the normalization applied. The layer is parameterized by the normalization parameters $Norm$, weight matrix W, and bias vector b.

The inputs to our model are the spatial region and the two time-dependent sequences. Our neural network first embeds each of the three inputs utilizing the feed-forward layer, where σ is Relu and Norm is batch normalization. For each input type j, an independent layer is learned, parameterized by $Norm_j$, W_j, and b_j. For the inputs consisting of a sequence, the embedding layer is applied element-wise to produce an array of embeddings $(h_{j1}, ..., h_{jT})$. $h_{ji} \in \mathbb{R}^h$ where h is the embedding size.

Following the embedding step, we utilize the attention mechanism to condense the two sequences of embeddings into fixed-length representations, which is the structure required by our model. The attention mechanism weighs the importance of each embedding in the sequence and ultimately returns a weighted average of the embeddings. For both arrays $(h_{j1}, ..., h_{jT})$, we concatenate the embedding of the region of interest h_{x_i} to produce a new sequence $([h_{j1}; h_{x_i}], ..., [h_{jT}; h_{x_i}])$. This new sequence is fed through another feed-froward layer element-wise, where σ is linear and there is no normalization, producing a scalar output denoted e_{ji} for embedding h_{ji}. We then utilize the softmax function to map the importance values e_{ji} to proper probabilities a_{ji}.

$$a_i = \frac{exp(e_i)}{\sum_{k=1}^{T} exp(e_k)}$$

We utilize these attention values to perform a weighted average of the embeddings in each sequence to produce a final embedding.

$$H = \sum_{i=1}^{T} a_i h_i$$

This mechanism allows the model to learn the relative importance of each of the embeddings conditioned on the embedding of the region of interest. After generating the embedding for both sequences, we concatenate the embeddings for all three inputs to form the final representation. This is then fed through a final feed-forward layer where σ is linear and there is no normalization, to produce scalar predictions for the mass balance $\hat{y}_i$. We train our model using the mean-

squared error loss function.

$$L = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Our model is trained on historical data from 1997 to 2015, wherein we extract approximately 10% of the data to serve as the validation set for determining when training should be cut off. To account for random variation due to the initialization of parameters, we re-initialize the model five times and select the model with the smallest validation error. Using cross-validation, we measure the test error for each of the 25 sites for 2015 to get our final modeling error metric.

Gaussian Process

One drawback of solely relying on a neural network for predicting mass balance is the lack of a straightforward approach to derive confidence intervals for said predictions. Having these intervals would provide meaningful insight into how model confidence varies depending on where the prediction is being made. To this end, we incorporate a Gaussian process on top of our neural network to generate these intervals. Prior work has shown potential for this approach.

We denote the Gaussian process as $f(x) \sim GP(m(x), k(x, x^{'}))$, where $m(x)$ is the mean function and $k(x, x^{'})$ is the kernel. We utilize a linear mean function parameterized by the weight matrix W and bias vector b of the final feed-forward layer of our neural network. Thus, the mean function is linear with respect to the stacked embedding representation h_i produced by our model preceding the last layer. In this way, we directly incorporate information learned by our neural network.

$$m(x) = Wh_i^T + b$$

We designed our kernel to utilize the spatio-temporal structure of our data. Based on prior work in this space, we use the Matern kernel (Matern 5/2 kernel). To incorporate spatial along with temporal information we utilize a multiplicative kernel, where the individual kernels model time (year) and space (longitude, latitude, elevation) independently.

$$k(x, x^{'}) = Matern_{spat} \ * \ Matern_{temp}$$

where

$$Matern_x = \sigma_x^2(1 + \sqrt{5}r_x + 5/3r_x^2)exp(-\sqrt{5}r_x)$$

where

$$r_x = \frac{||h_x - h_x^{'}||_2^2}{2\ell_x^2}$$

Here σ denotes the variance parameter and ℓ denotes the lengthscale parameter.

We employ Hamiltonian Monte Carlo to find the optimal values for each of our four parameters σ_{spat}, ℓ_{spat}, σ_{temp}, and ℓ_{temp}. To do so, we first initialize the model parameters to the maximum likelihood solution. We then fix Gamma priors to each of the parameters. Finally, we run the chain until convergence, using autocorrelation to determine whether the chain has properly converged.

We found that the use of the Gaussian process improves the modeling performance for the winter mass balance predictions, while slightly decreasing the performance for the summer mass balance predictions. Still, we believe that the addition of the generated confidence intervals more than makes up for the small decrease in modeling performance.