U.S. BUREAU OF RECLAMATION'S

WATER SUPPLY FORECASTING RODEO

Contestant Writeup: @mmiron
(Murray Miron for SRLLC)

March 28, 2024

ABSTRACT

While nearly every line of the program was eventually changed, the SRLLC-authored code used to win several months in a (year long) 2021 Bureau of Reclamation challenge was used as a starting point for the Water Supply Forecasting Rodeo. My Final stage submission, which simulated prediction of annual streamflow volume six or fewer months in advance, scored 30.6747 by the pinball loss metric (very roughly equivalent to half of the mean absolute error) in units of thousand-acre-feet. Sources of data include the Applied Climate Information System (ACIS), the Snow Telemetry Network (SNOTEL), and antecedent monthly flow volume for a given site. To make predictions, the input values are aggregated in monthly time periods; they are then preprocessed, stacked together, and fed through a trained TensorFlow model. The output of said model represents the predicted seasonal volume of streamflow, with a confidence interval, for a given year and site. The architecture of the artificial neural network includes an irregular implementation of a standard self-attention mechanism, and prediction-specific feature importance rankings. The trained weight files are also less than a megabyte in size, making it small enough to run in even an embedded environment.

Contents

List of Tables

List of Figures

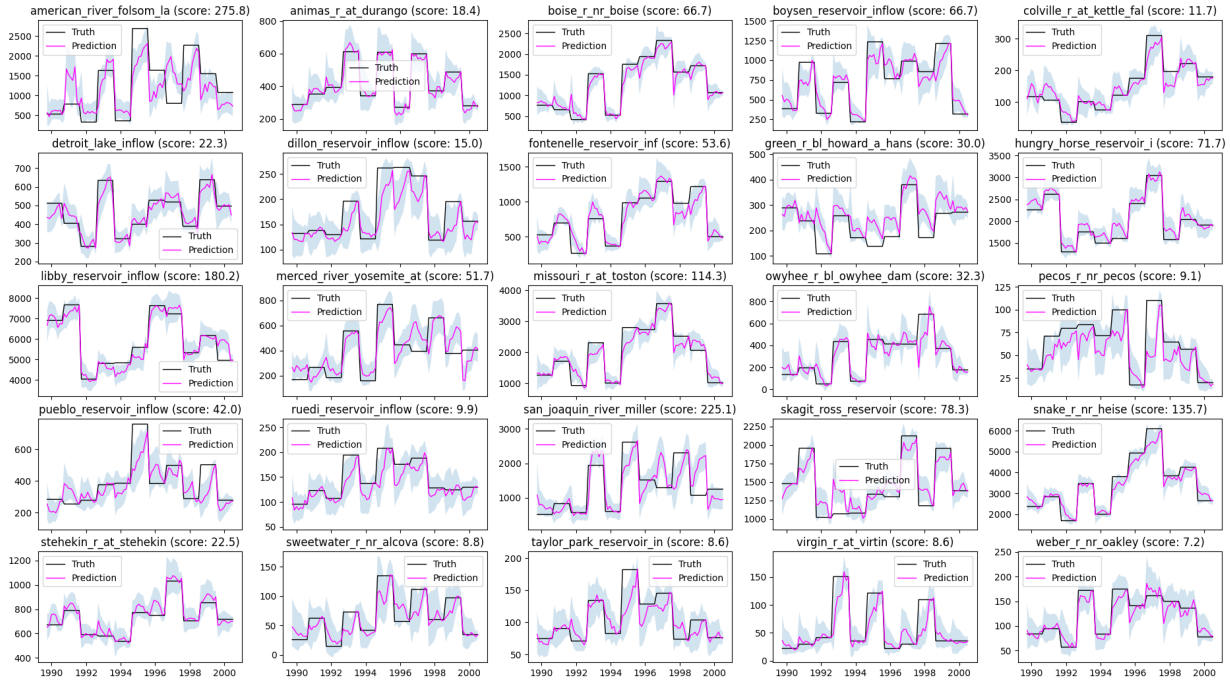

Figure 1: Trained model predictions made throughout the year – 4 per month – for annual streamflow, with a confidence interval of 80%, for years 1990-2000.

1 Technical Approach

1.1 Algorithm and Architecture Selection

The model input consists of eight channels of length-10 arrays, and the three values output by the model are used as the predictions without further processing. The algorithm and model used for the Final stage submission expect no explicit time feature(s), instead relying on a rigid structure to communicate it. This means that the only time information available to the model when making a prediction is the structure of the input. An illustration of what is meant can be found in figures 4 and 5, as well as further details in the training section.

The model architecture utilizes a self-attention mechanism. This provides a means of indicating which values were integral in issuing a specific prediction. Note that this is in contrast to a gradient boosting decision tree algorithm, which only provides importance rankings for the construction of the model itself (i.e. one generic ranking for all predictions, not feature importance for a specific prediction). As can be seen from figure 3, the attention blocks are responsible for providing the rest of the model with its input data. If this were not the case, the model would have no means of reporting which features were important for a given prediction (or more to the point, the output of the attention blocks would be irrelevant).

The unusual application of self-attention will be covered in more detail in the Explainability report, but broadly: the attention and feed forward blocks serve to amplify or diminish data within channels. The structure of the output of these attention blocks is identical to the structure of their input, and by contrasting said output with the unprocessed input given to these blocks, weights can be inferred. See figure 2 for a preliminary example (note that implementation details are still being worked out, and may change).

The Final stage submission has 94,069 trainable network parameters in total, making the weight files used for predictions an extremely lean 367.46 kilobytes. Two noise layers are also utilized during training, but have no effect during prediction. Further details on the architecture can be found in the training section.

A more Transformer-like time series approach – that is, one in which timestamps themselves are another input feature – was experimented with, but initial results were poor. In the absence of any tangible need to adhere to the flexibility of making time a simple feature, that method, as well as the many Transformer architectures designed for time series

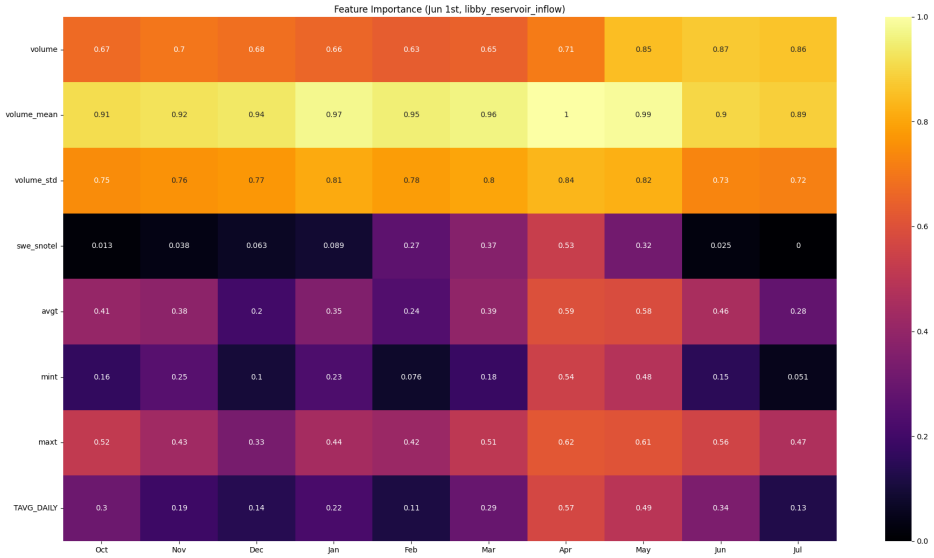

Figure 2: Preliminary feature importance for a Libby Reservoir Inflow prediction

forecasting (Temporal Fusion Transformer, Autoformer, and PatchTST were among those experimented with), were abandoned.

Also tried, but ultimately abandoned, was a classical time series forecasting method that simply predicted monthly values. Both autoregressive [1] and long forecasting horizon methods [2] were attempted. The appropriate months for a given site were then summed to arrive at final predictions; however, when the model output was reduced from multiple months to only one (seasonal) prediction value, it consistently scored better on the validation set.

An XGBoost model was attempted as well, but was greatly outperformed – according to final score, not training time – by TensorFlow models.

See figure 3 for a diagram of the TensorFlow model architecture; note that only one attention block, including one feed forward strip and one subsequent convolutional block, are shown for brevity. In practice, multiple blocks are used in succession. Their output is then passed through a linear layer (i.e. dense "top" layer), and the result is the final prediction.

1.2 Data Sources and Feature Engineering

There were a number of approved data sources experimented with that resulted in decreased prediction accuracy, despite intuition to the contrary. My conclusion is that there are many data sources that would improve prediction accuracy, but *not for all sites*, and as such, their inclusion in the general case worsens accuracy overall (due to the scoring formula being an average of performance over 26 sites).

In the process of reaching this conclusion, there was a great deal of experimentation that was performed. The average lifetime of water molecules in water cycle reservoirs was used to estimate which data sources were of influence within a given water year; see table 1 [1].

Excluding all sources with residence times of a year or more, this meant snowpack and precipitation were the only immediately influential sources of water. During my research it became clear that temperature also plays a role in streamflow, above and beyond its effects on snowpack, rainfall, and evapotranspiration.

[1] Autoregressive is used herein to refer to predicting one month at a time, before using that prediction to issue predictions for additional months.

[2] By long horizon, what's meant is the practice of predicting a number of months simultaneously (or in one forward pass through the model).

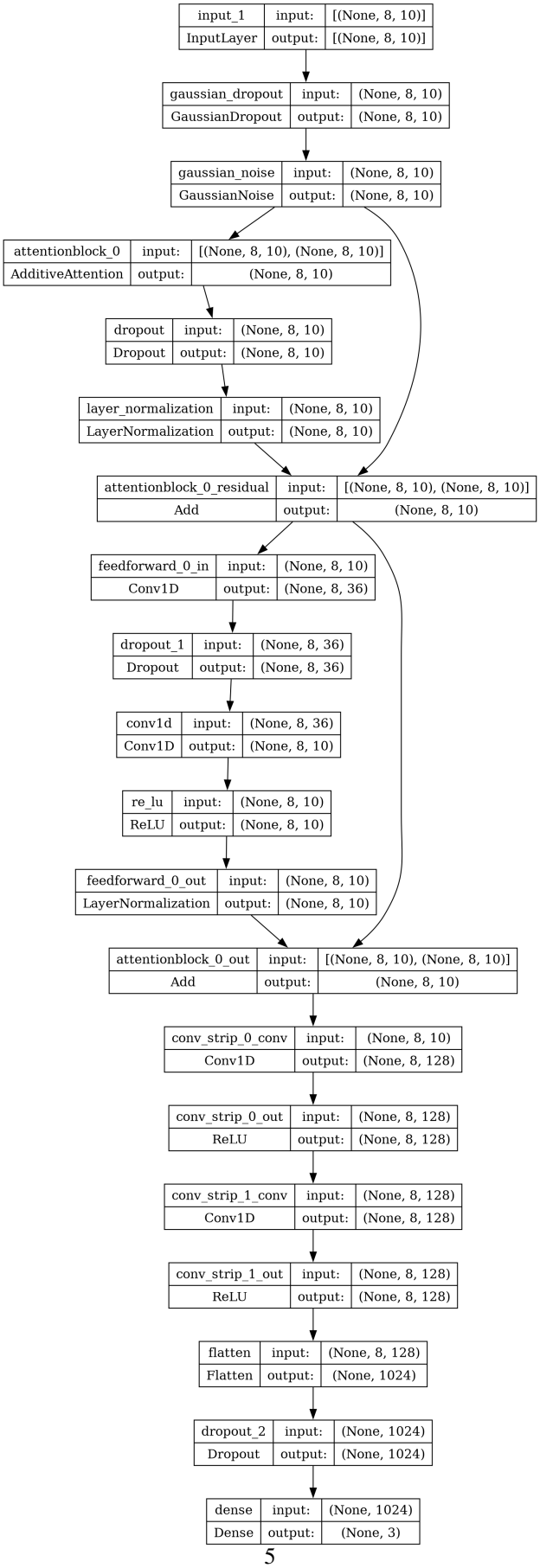

Figure 3: My TensorFlow model architecture (a simplified single-block version).

Reservoir	Average Residence Time
Glaciers	20 to 100 years
Seasonal Snow Cover	2 to 6 months
Soil Moisture	1 to 2 months
Groundwater: Shallow	100 to 200 years
Groundwater: Deep	10,000 years
Lakes	50 to 100 years
Rivers	2 to 6 months

Table 1: Typical residence times of water found in various reservoirs [1].

Code was written to collect and/or use the following data sources:

- The University of Arizona UA/SWANN snow water equivalent measurements.
- The California Data Exchange Center (CDEC) Snow Sensor Network.
- The NRCS Snow Telemetry (SNOTEL) network.
- The Snow Data Assimilation System (SNODAS).
- NASA's Moderate Resolution Imaging Spectroradiometer (MODIS) satellite telemetry (Vegetation Indices and Snow Cover products).
- Southern Oscillation Index (SOI) measurements.
- Pacific Decadal Oscillation Index (PDO) measurements.
- The NOAA Regional Climate Centers Applied Climate Information System (RCC-ACIS).
- The U.S. Geological Survey's (USGS) streamgage measurements.
- The U.S. Bureau of Reclamation's (USBR) naturalized inflow measurements.

Faced with limited time available to explore permutations of input data, I eventually decided – via substantial trial and error, meaning that the inclusion of all other attempted data sources diminished the solution's performance – to focus on the following features, each of which constitutes its own "channel" of input to the model:

- Antecedent streamflow values (per month).
- Mean annual streamflow value (per site).
- Standard deviation of annual streamflow value (per site).
- SNOTEL network snow water equivalent measurements (per month, and within 64,373.8 meters – approximately 40 miles – of a site's drainage basin).
- Average temperature (per month, a gridded value provided by ACIS for the coordinates of a measurement site).
- Minimum temperature (per month, gridded value for the site, as above).
- Maximum temperature (per month, gridded value, as above).
- Average temperature (per month, provided by the SNOTEL network and including a 40 mile radius around a measurement site's coordinates).

Examples of a raw input matrix and the same, but annotated, input matrix are given in figures 4 and 5, respectively.

Each "channel" is composed of ten scalar values, all of which are the result of aggregating data for a calendar month. The channels are combined via simple stacking and fed to the model as a two dimensional matrix of shape $(8, 10)$. Values that are missing (unavailable) or in the future are assigned a simple value of zero.

1.2.1 Enumeration of data channels

- The monthly values are what the annual number is ultimately composed of, which serves to explain the usefulness of its data channel. Only months within a given water year [3] are included (and among those, only values that would have been available prior to a given issue date), preserving the treatment of water years as independent observations.

[3] As used herein, a water year is defined as beginning on October 1st and ending in June, July, or August of the following year.

Figure 4: Raw model input - a training sample for Libby Reservoir Inflow simulating a prediction with an issue date in March.

```
[[3.41898987e+02 5.04631012e+02 3.47755005e+02 2.68312012e+02
  2.17893005e+02 0.00000000e+00 0.00000000e+00 0.00000000e+00
  0.00000000e+00 0.00000000e+00]
 [5.62900465e+03 5.62900465e+03 5.62900465e+03 5.62900465e+03
  5.62900465e+03 0.00000000e+00 0.00000000e+00 0.00000000e+00
  0.00000000e+00 0.00000000e+00]
 [1.28441512e+03 1.28441512e+03 1.28441512e+03 1.28441512e+03
  1.28441512e+03 0.00000000e+00 0.00000000e+00 0.00000000e+00
  0.00000000e+00 0.00000000e+00]
 [1.38351254e-01 3.54074074e+00 7.45304659e+00 1.43437276e+01
  2.38353175e+01 0.00000000e+00 0.00000000e+00 0.00000000e+00
  0.00000000e+00 0.00000000e+00]
 [4.18437500e+01 3.38125000e+01 2.56875000e+01 2.77031250e+01
  2.46093750e+01 0.00000000e+00 0.00000000e+00 0.00000000e+00
  0.00000000e+00 0.00000000e+00]
 [3.00066410e+01 2.66738280e+01 1.99273450e+01 2.10945320e+01
  1.40281260e+01 0.00000000e+00 0.00000000e+00 0.00000000e+00
  0.00000000e+00 0.00000000e+00]
 [5.34171870e+01 4.00015640e+01 3.09778330e+01 3.39529300e+01
  3.53363270e+01 0.00000000e+00 0.00000000e+00 0.00000000e+00
  0.00000000e+00 0.00000000e+00]
 [3.32115207e+01 2.61090476e+01 2.05253456e+01 2.04064516e+01
  1.71698980e+01 0.00000000e+00 0.00000000e+00 0.00000000e+00
  0.00000000e+00 0.00000000e+00]]
```

- Including the mean and standard deviation for a given measurement site ensures that the model can learn to predict confidence intervals. These historical numbers are the result of multiple water years, however, their calculation explicitly excludes any values that fall within test year(s).
- The fourth channel of input used by the solution is a measure of snow water equivalence, [4] as provided by the SNOTEL network. SNOTEL measurement sites within a 40 mile radius of a given forecast site's drainage basin (64,373.8 meters) are used. This is in contrast to my Hindcast stage submission, which used CDEC and UA/SWANN snowpack measurements as well.
- The fifth, sixth, and seventh channels are temperature measurements provided by the Applied Climate Information System (ACIS). These are gridded values downloaded from the grid2.rcc-acis.org server. The input provided to the grid2.rcc-acis.org endpoint when requesting values are the longitude and latitude for a given site, as taken from the site metadata CSV file provided by the USBR. Only data for months prior to the month a prediction's issue date is within, and then only for months within a given prediction date's water year, are used.
- The eighth channel is the average temperature provided by the SNOTEL automated measurement equipment around a 40-mile-buffered drainage basin area, as described for the fourth channel (snowpack measurements) above. The months used are identical. This channel primarily provides redundancy, since real world data rarely plays nice and may be erroneous or unavailable.

1.3 Uncertainty Quantification

The model itself directly predicts the predefined quantiles. Of the three numbers output by the artificial neural network, the first is the prediction for the 0.1 quantile. The second is the prediction for the 0.5 quantile. The third is the prediction for the 0.9 quantile. This is a result of the definition of the "tau" parameter in the loss function used during training (see figure 8). Any arbitrary real number (representable by a floating point approximation) could have been used, if a different confidence interval – or indeed, multiple confidence intervals – were desired.

[4] As used herein, SWE is a number that indicates how much water a given amount of snow will become when melted.

Figure 5: Raw model input - annotated version of figure 4.

	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May	Jun	Jul
volume	341.898987	504.631012	347.755005	268.312012	217.893005	0.0	0.0	0.0	0.0	0.0
volume_mean	5629.004652	5629.004652	5629.004652	5629.004652	5629.004652	0.0	0.0	0.0	0.0	0.0
volume_std	1284.415120	1284.415120	1284.415120	1284.415120	1284.415120	0.0	0.0	0.0	0.0	0.0
swe_snotel	0.138351	3.540741	7.453047	14.343728	23.835317	0.0	0.0	0.0	0.0	0.0
avgt	41.843750	33.812500	25.687500	27.703125	24.609375	0.0	0.0	0.0	0.0	0.0
mint	30.006641	26.673828	19.927345	21.094532	14.028126	0.0	0.0	0.0	0.0	0.0
maxt	53.417187	40.001564	30.977833	33.952930	35.336327	0.0	0.0	0.0	0.0	0.0
TAVG_DAILY	33.211521	26.109048	20.525346	20.406452	17.169898	0.0	0.0	0.0	0.0	0.0

Figure 6: Raw model input - the data from figure 5, permuted to simulate ACIS average temperature unavailability (missing value emphasized by added asterisks).

	Oct	Nov	Dec	Jan	Feb	Mar	Apr	May	Jun	Jul
volume	341.898987	504.631012	347.755005	268.312012	217.893005	0.0	0.0	0.0	0.0	0.0
volume_mean	5629.004652	5629.004652	5629.004652	5629.004652	5629.004652	0.0	0.0	0.0	0.0	0.0
volume_std	1284.415120	1284.415120	1284.415120	1284.415120	1284.415120	0.0	0.0	0.0	0.0	0.0
swe_snotel	0.138351	3.540741	7.453047	14.343728	23.835317	0.0	0.0	0.0	0.0	0.0
avgt	41.843750	33.812500	25.687500	27.703125	* 0.0 *	0.0	0.0	0.0	0.0	0.0
mint	30.006641	26.673828	19.927345	21.094532	14.028126	0.0	0.0	0.0	0.0	0.0
maxt	53.417187	40.001564	30.977833	33.952930	35.336327	0.0	0.0	0.0	0.0	0.0
TAVG_DAILY	33.211521	26.109048	20.525346	20.406452	17.169898	0.0	0.0	0.0	0.0	0.0

1.4 Training and Evaluation Process

1.4.1 Construction of the training samples

In total, 1.8 million training samples were utilized during the training process. This number does not take into account the random jittering provided by the first two noise layers of the model, which effectively multiply the number of samples the model was exposed to during training.

The supplementary data provided by the challenge organizers was augmented before use. The augmentation process consisted of adding the ACIS and SNOTEL data to the supplementary sites. Additionally, random values were replaced with zeroes to simulate lagging or missing data (see figure 6).

To add the RCC-ACIS data (minimum, maximum, and average temperature values) to the supplementary sites, I:

1. Collected the geographic coordinates for each supplementary site from the provided supplementary_metadata.csv file.
2. Downloaded as much historical temperature data as is available from grid2.rcc-acis.org, providing the supplementary site coordinates as input to the server endpoint.

To add the SNOTEL data (snow water equivalent and average temperature) to the supplementary sites, I:

1. Added all supplementary sites to the list of applicable SNOTEL stations. To do this, I performed a spatial join between the coordinates found in supplementary-metadata.csv and the sites_to_snotel_stations.csv file (provided by the organizers and the wsfr_download package).
2. Used an approximately 40 square mile buffering radius when performing the spatial join described above.

I combined the resulting data – that is, the augmented historical data for the hundreds of supplementary sites – with the augmented data for the 26 sites that are considered in the scoring formula. For all several hundred sites, training samples were constructed that represented each possible month within a water year that a prediction could be made

Figure 7: Raw model input - the target for the sample in figure 5 (5,596.5 thousand-acre-feet of seasonal streamflow).

```
[ 5596.5, 5596.5, 5596.5 ]
```

Figure 8: My Python code to calculate the loss during training.

```
import tensorflow as tf

def quantile_score(y_true: tf.Tensor, y_pred: tf.Tensor, tau: float) -> tf.Tensor:
    zero = tf.cast(0, tf.float32)
    one = tf.cast(1, tf.float32)

    term1 = tf.math.maximum(y_true - y_pred, zero)
    term2 = tf.math.maximum(y_pred - y_true, zero)
    return (tau * term1 + (one - tau) * term2)

def contest_loss(y_true: tf.Tensor, y_pred: tf.Tensor) -> float:
    scores = list()

    for i in range(0, y_pred.shape[-1], 3):
        scores.append( quantile_score(y_true[:,i], y_pred[:,i], tau = 0.1) )
        scores.append( quantile_score(y_true[:,i+1], y_pred[:,i+1], tau = 0.5) )
        scores.append( quantile_score(y_true[:,i+2], y_pred[:,i+2], tau = 0.9) )

    return tf.math.reduce_mean(scores)
```

during. For example, for a given water year and a given site, an 8-by-10 matrix for each month was used: to simulate a prediction being made in January, all the values past the first 3 positions would be zeroes; each successive month was represented by an additional value added (e.g. February would have any values past position 4 zeroed out), until finally, the entire input matrix would be filled with non-zero values (to simulate a prediction being made in June). To help illustrate my meaning, the training sample seen in figure 5 simulates a prediction being made in March.

I also took the resulting matrices, constructed as described in the immediately preceding paragraph, and randomly permuted them by lagging behind RCC-ACIS and SNOTEL columns; this was done to simulate unavailable data points at the time a prediction is issued. An example is shown in figure 6.

1.4.2 Construction of the training targets and loss metric

The custom loss function calculated the loss using the contest scoring formula. Each quantile's contribution to the averaged score was calculated using a position in the target array, i.e. (0.1, 0.5, 0.9). My simplified Python/TensorFlow code to do so can be seen in figure 8.

The target arrays were of length 3, and filled with the repeated seasonal value for a given site in a given year. An illustration can be seen in figure 7. Length 3 arrays were used for the sake of convenience, as it made implementing the custom loss function straight forward. The three numbers output by the model are then clipped to be between 0 and infinity, but are otherwise used as the predictions without further post-processing.

1.4.3 Validation set construction

During development, a simple holdout set of 11 years (1990 through 2000, inclusive) was used as the validation set. For the Final stage submission, I used the designated years (2004 through 2023) iteratively in the cross-validation process, instead.

1.4.4 Cross-validation scores for the Final stage submisson

Certain years are significantly more difficult for the model to predict than others. This can be seen in table 2 (the year with the best performance is shown in bold). For results of leave-one-out cross-validation over the entire dataset, i.e. all primary sites as well as all supplementary sites, see table 3. The two results are discussed in section 4.

Test Year	Averaged Mean Quantile Loss
2004	**39.1**
2005	51.4
2006	72.0
2007	60.5
2008	66.4
2009	47.7
2010	46.0
2011	99.1
2012	69.3
2013	68.2
2014	44.6
2015	58.3
2016	48.8
2017	99.5
2018	50.4
2019	78.4
2020	56.7
2021	44.2
2022	72.4
2023	77.4

Table 2: Cross-validation - averaged mean quantile loss over the 26 primary sites.

2 Solution Differences by Stage

Both major and minor changes to my approach were made throughout all stages of the contest. Not all of these changes resulted in quantitative gains, but several were responsible for drastic improvements in prediction accuracy. As such, I note the most significant aspects by stage below.

2.1 Hindcast Stage Solution

Initially I approached my solution with a classical time series forecasting method, and attempted to predict monthly streamflow. A substantial increase in prediction accuracy resulted when the model was changed to predict only annual streamflow.

2.2 Forecast Stage Solution

There were two dramatic gains made between the Hindcast and Forecast stage: refinements to my model architecture, and the realization that I had made a mistake when concluding that using the supplementary site data worsened the model's accuracy. At the time I made this observation, all data channels except the mean, standard deviation, and antecedent monthly flow measurements were zeroed out when fed to the model. After implementing the code required to augment the supplementary site data with the same data sources I was using for the 26 scored sites, and including it in my training data, I observed the greatly improved accuracy one would expect when using a much larger training set. My conclusion is that fitting the model to data with several channels containing nothing but zeros, effectively "unlearned" what the training process "taught" the model when fitting it to samples containing useful data in those channels.

The Forecast stage submission uses an ensemble of four models, all of which follow the basic architecture described herein. Two of the models are greatly enlarged, and two are smaller versions. After the Forecast deadline passed, and upon further experimentation, I concluded that enlarging the neural networks ultimately shows no appreciable gain in prediction accuracy.

2.3 Final Stage Solution

Regrettably, my Forecast stage performance was hindered by a failure to sufficiently compensate for missing or otherwise corrupted data. I altered the training process to selectively corrupt training samples, as previously described, and performance improved commensurately. The cross-validation scores are also the result of a single model without any ensembling.

3 The Solution as Software

Experimenting with various methods of solving a problem is always problematic for the design of software, as the software must constantly be redeveloped and/or redesigned to suit the current experiment requirements. As one might imagine, these constant design changes and the extremely fast pace do not lend themselves well to clean code. To accommodate this process, I refined my approach as the contest progressed, and eventually produced an extremely flexible package of just over 5,000 lines of code.

At the time of this writing, the software solution described herein is implemented in a modular, object oriented fashion. To add a data source, all that needs to be done is inheriting from the DataHelper class and implementing a single virtual method in a new file, as well as adding the name of the resulting data channel to the list of existing input sources. So long as the added channel conforms to the same structure described in earlier sections, i.e. contains data that does not cross water year boundaries and is aggregated by month, the architecture of the artificial neural network will be modified to accommodate the additional input. Note that this includes consideration of the new input channel when reporting feature importance. Any impact on prediction performance when using the new combination of data sources can then be evaluated by retraining and retesting the model, which can now be performed by two commands: one to select the optimal weights for the modified architecture (training), and one to test those weights (scoring).

4 Generality of the Solution

Leave-one-out cross-validation was performed on all 612 sites (the 586 supplementary sites as well as the 26 primary sites). As can be seen from table 3, model accuracy is not restricted to the 26 primary sites, and prediction accuracy is largely the same for all 612 locations. The similar performance seen when the model is used to make predictions for sites outside those considered in the contest scoring formula strongly suggests wide applicability. Note, however, that in five cases – i.e. less than 1% of cases – the mean quantile loss for specific sites is over 1,000 in 2004 (these sites are *not* among those scored in the contest). At the time of this writing, no cause for these poor scores is apparent. See table 4.

5 Machine Specifications

The models were trained on a Linux machine with an Intel Core i7-7700K CPU and NVIDIA GeForce RTX 3090 GPU, using Linux v6.5.9, Python v3.11.5, TensorFlow 2.15.0.post1, and CUDA 12.1.105.

An early stopping patience of 1,500 epochs was used, and the average number of training epochs throughout the cross-validation process was approximately 4,000 per year. On the development machine, the entire 20 year cross-validation process takes about half a day to finish training (i.e. all 80,000 epochs, or 4,000 epochs per year * 20 years). Inference is substantially faster, and predictions can be issued in less than a second: the 728 predictions for a single year's cross-validation process take half a minute on the development machine, for example (4 predictions per month * 7 months per year * 26 sites). The prediction process could be optimized and would take a fraction of this time, if a compiled language were used in place of Python; but for the purposes of the contest, there was no need.

Python packages of major importance include TensorFlow (TensorFlow-CUDA), Pandas, NumPy, and matplotlib.

6 References

References

[1] Pidwirny, M. (2006). "The Hydrologic Cycle". Fundamentals of Physical Geography, 2nd Edition. https://web.archive.org/web/20160126072955/http://www.physicalgeography.net/fundamentals/8b.html

Test Year	Averaged Mean Quantile Loss	Interval Coverage
2004	**38.16**	**.88**
2005	50.60	.87
2006	51.98	.89
2007	51.84	.84
2008	52.71	.89
2009	46.26	.93
2010	43.92	.87
2011	85.43	.81
2012	65.63	.91
2013	53.79	.89
2014	45.52	.85
2015	48.05	.83
2016	44.56	.86
2017	60.31	.87
2018	49.69	.84
2019	69.72	.89
2020	76.61	.79
2021	45.90	.89
2022	99.04	.83
2023	55.19	.85

Table 3: Cross-validation - averaged mean quantile loss over all 612 sites.

Test Year	Site	Mean Quantile Loss
2004	Yukon R nr Stevens Village	1,451.41
2004	Yukon R at Eagle	1,323.96
2004	Columbia R bl Rock Island Dam-NWS	1,213.11
2004	Columbia R at Grand Coulee-NWS	1,200.61
2004	Columbia R at The Dalles-NWS	1,041.74

Table 4: Cross-validation - sites with mean quantile losses over 1,000 in 2004.