Using Transformers to Impute Missing Data

By Zhengyuan Cui and Cameron Fen

Problem Description:

Current Population Survey (CPS) is a dataset that measures employment behavior among individuals, for example, employment status, wages, etc. This dataset surveys approximately 60000 since the 1960s when people are added and dropped off the survey on a rolling basis. This data is used to calculate the national employment statistics as well as the unemployment rate. However, as this data is survey based, there are missing data as people may not answer their survey. This data surveys people once a month for four months, then waits 8 months and then resurveys people for 4 months 13-16 months after the beginning of the sample. As such, there are imputation needs both for survey non-respondents as well as by model design to filling in missing data over the 8 months when the survey is not active. Current methods used by the census are not very sophisticated and involve using a technique called hot-decking which is essentially 1-nearest neighbors on a stratified sampled. More detail of the method will be discussed in the related works section.

Our work shows that one can improve the imputation of this data by using a transformer model that has been taken from the NLP literature. The benefits of this imputation are two-fold. This method will improve the imputation of census data which is a direct benefit to the Census Bureau which imputes missing data before aggregating the data to form national statistics. This imputation is necessary because the inputs need to have a representative sample of the US population in the survey and certain subpopulations may be more prone to non-response. A second benefit is that this is a proof of concept for an imputation method that can be more widely used across the social sciences especially when performing inference on large datasets with a timeseries/width dimension in addition to a cross section dimension. These techniques are necessary as often one must resort to less accurate imputation techniques which will lead to more biased outcomes and conclusions drawn from the data.

An example of how our method works. First, we attempt to impute the hours worked data. Suppose there are 8 data points listing hours worked over a 16-month period. The input is this data along with a positional encoding. Say the data is 40, 35, 45, 40, M, M, M, M, M, M, M, M, 50, 60, 45, 50, where the M indicates missing data. Say we wanted to infer what the M's should be. Our transformer would take this data and output values M, M, M, M, 45, 40, 55, 50, 55, 60, 40, 45, M, M, M, M. With the M denoting output that is ignored since that data is already available. Here there is already another advantage over the traditional way of hot-decking. Hot-decking chooses a nearest neighbor, however in this example the masked data is from month 5 to month 12, which is missing for all data points and so hot-decking cannot give us information on this data. However, our method can still give use insight leveraging the transformers ability to generalize across positional encodings.

References/Related Work:

The census' current methodology for filling in missing data is highlighted in this white paper (US Census Bureau, No Date). In particular, the census uses the “hot decking” approach which is essentially a one-nearest neighbors search to impute values. A survey of the literature on census imputation is given in (Rubin, 1983). The basic intuition behind hot-decking is to take the nearest neighbor of a missing data point and use the value of the nearest neighbor to impute the missing value. Another class of methods that we don't implement is the use of explicit methods like the Heckman selection model. The methodology is described in depth in (David et al., 1986), but the basic idea is to use a statistical model like a linear regression to impute missing values at a given location.

Both types of methods have their own weaknesses. As pointed out by Rubin, the hot-decking approach and related methods like k-nearest neighbors will always underestimate the standard deviation of the relevant imputation. Marginalization by sampling from the imputation distribution is one technique which allows for efficient estimation of a second stage problem (Rai, 2020). The explicit models like the Heckman Selection approach allow for estimating of this missing data by sampling from the imputation distribution, but the hot-decking approach even if one averages across k-nearest neighbors always underestimates it. However, the problem with explicit approaches is they cannot utilize information gained from other observations. What I mean by that is that assume the first month has a nonresponse: M, 25,30,30,M,M,M…, this would require a different model than 25, M, 30, 30, M, M, M… because the mask is in a different location and it requires different weights on all the surrounding datapoints.

The transformer combines the best of both approaches as you have one model that can model missing data among each position in a joint fashion, but is also a explicit approach in the sense that the transformer can provide uncertainty estimates for each estimate allowing efficient (in the statistical sense) estimation after imputation by sampling from the imputation distribution. However due to the time constraint and the consensus that hot-decking probably has better performance, we mainly compare our model to k-nearest neighbor approaches and mean and median baselines rather than other explicit models.

Methodology

We use data from the Census Bureau downloaded from IPUMS. We attempt to impute randomly masked hours worked data based on the Census Bureau survey. Just as a reminder, this data has 4 months of reported data, 8 months of unreported data, and 4 more months of reported data. We only look at data that has no missing values (and we mask one value), we also remove some nonsensical reporting like data that has negative hours worked or more than 168 hours worked in a week (ie greater than 24*7). We attempted to use other data like employment status, state, date of survey etc to improve forecasting, but we determined the only factors that improve performance are the actual seven other hours worked data as well as the positional encodings of all data relative to the 16 month survey window.

Our model uses just an encoder of 6 stacked transformers with a dense net with 128 dimensional hidden layers. We used Adam optimizer and did not use dropout nor weight decay. Since any regularization method seemed to decrease performance, we suspect we are still underfitting but increasing parameter size didn't seem to improve performance either. For our training and validation set, the best performing results was when we masked every row at a random location. This improved upon random masking which ignored the fact that some entries (ie 5-12) are always masked. We also attempted to use a encoder-decoder architecture but unsurprisingly as this doesn't involve return a different output sequence compared to an input sequence this didn't improve performance. Perhaps more surprisingly, when we attempted to mask only the hours worked data but allowed the positional encoding to remain unmasked in a decoder framework, this reduced performance. We also attempted to change to random masking at each iteration which also decreased performance.

We originally started without positional encoding and performance was quite bad. Adding positional encoding improved performance from around RMSE of 9 or 10 to about 5.25. Moving to Adam improved the model to about 4.9. Adding masking on every row and not just allowing masking to be random improved performance to 4.83 on our validation set which was our best result.

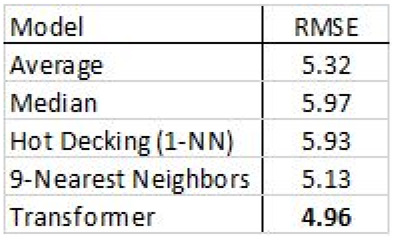
Experiments

The dataset we used came from the CPS. After filtering out implausible survey results as well as surveys that were missing responses, we had 845547 people in our dataset from 1994-2020 each with 8 responses. We put 700000 in our training set, 100000 in our validation set and 40000 in our test set, choosing to ignore the remaining 5547 to keep the data at round numbers.

We evaluate both model performance and training loss using mean squared error. We initially tried breaking the data into 169 different buckets-ie 0 hours a week worked all the way to 168 hours a week worked. Then trained to model based on cross entropy loss. Unsurprisingly, the mean squared error performance of this model was not as good as when we trained the model using mean squared error. However this is one way to get distributional results and allow for sampling of imputation distribution which leads to efficient second step estimation. Likewise, another approach which we will implement in the future is to do amortized inference where the model returns both a mean and a standard deviation for each masked output and you maximize the likelihood of the data occurring.

The baseline models we use include hot-decking both 1-NN and 9-NN, which was the best performing nearest neighbors on the validation set. Then we also report mean imputation and median imputation where we use the mean or median respectively of the other 7 values and use that to impute the masked value. We haven't implemented more state space models like a Kalman smoother or a bidirectional RNN, partially because even though we think performance will be good these methods aren't used in practice. We want to implement something like a Heckman Selection model as a baseline, but we didn't have time this semester.

In order to get a more accurate test set evaluation, we masked out all 8 values on the test set, but multiplied the size of our data set by eight so each mask still has 7 other unmasked data to make inference under. We do this to make a more persuasive argument that our test set is not just a subset of the population of possible imputed value but the whole population over that time period leaving less room for evaluation noise. We think doing this same thing in our training data will also improve performance of our model by some amount.



Model	RMSE
Average	5.32
Median	5.97
Hot Decking (1-NN)	5.93
9-Nearest Neighbors	5.13
Transformer	**4.96**

Above is a chart that compares our transformer model with the baselines on our test set. Our transformer model can perform efficient second step inference, output uncertainty estimates, model multiple missing values, as well as perform reasonable inference on values missing from the entire data set, all of which none of the above models (or even other explicit models like the heckman selection model which we have not implemented) can do. Despite this versatility, our transformer also outperforms all the baseline models shown in this table. While the outperformance is slight, we think we can improve this model with additional improvements.

For future work we want to also include data even if it is missing 1 or more survey responses for hours worked. We think adding additional data will improve the transformer model accuracy and make it more flexible in imputing when less than 7 data points are present. We also want to implement the amortized inference which would allow efficient second step inference as well as implement improvements to the transformer model like appending nearest neighbors to the input for the model to reference.

Bibliography

Bureau, US Census. “Imputation of Unreported Data Items.” The United States Census Bureau, 3 May 2016, www.census.gov/programs-surveys/cps/technical-documentation/methodology/imputation-of-unreported-data-items.html.

David, Martin, et al. "Alternative methods for CPS income imputation." *Journal of the American Statistical Association* 81.393 (1986): 29-41.

Rubin, Donald. "Imputing Income in the CPS: Comments on" Measures of Aggregate Labor Cost in the United States"." *The measurement of labor cost*. University of Chicago Press, 1983. 333-344.

Rai, Bhavna. “Imputing Missing Covariates Values in Nonlinear Models” 2020 https://drive.google.com/file/d/1GnbnnQSfiRsozBiXIwMitlUP1Ro1y9_U/view