Machine Learning for Graphs

Project Proposal

Martynas Vaznonis (2701013)
m.vaznonis@student.vu.nl

I have selected the paper "GraphCast: Learning skillful medium-range global weather forecasting" for the project I wish to extend. I have come up with a handful of possible extensions, though I am not perfectly pleased with any them. To arbitrate among them or suggest a better approach, I have contacted the professor assigned to the corresponding core topic but have yet to hear back from him. Therefore, I will list all of the possible extensions here, and go a little more in depth on just one. The possible extension ideas are as follows:

1. *Improving the encoder/decoder* - the encoder and the decoder only contain one layer each. Maybe two layers could improve the overall model by enabling it to encode nonlinear features and do likewise in decoding. But this is probably very unnecessary because of the deep processor GNN. I also considered implementing a different encoder/decoder for the different geographical regions. This could help as, e.g., the same features above an ocean and above a desert would likely contribute to different weather dynamics because of the way light is absorbed, the role of humidity, and so on. But this is, at least partially, solved in the original implementation by including force and constant features. Still, there might be some potential gain from geographically specific encoders/decoders, but they will likely add more complexity than they are worth.

2. *Extending the number of input states* - The authors claim that 2 input states are better than 1, but that 3 are too expensive for the little improvement they afford. My thinking was to learn based on the two most recent states X^t and X^{t-1}, same as in the original, but to also include a context state $C^{t-2-n:t-2}$ which aggregates information of n states from X^{t-2-n} to X^{t-2}. The authors do imply that the addition of a third state was beneficial, just ineffective. And a larger context would reduce the reliance on the Markov assumption which is not satisfied in such a system. The aggregation function could be learned while training GraphCast. Perhaps a simple transformer network could do.

3. *Combining machine learning weather prediction with numerical weather prediction* - The HRES model captures many of the important causal relations in modeling weather. Perhaps GraphCast can benefit from this by only modeling where the weather deviates from HRES's prediction. As is, GraphCast predicts by outputting change Y^t from the current state X^t, with $X^{t+1} = X^t + Y^t$. My idea is to simply change this prediction to $X^{t+1} = X^t + X^{t+1}_{HRES} + Y^t$, where X^{t+1}_{HRES} is the prediction from the numerical model HRES. This may free GraphCast to learn more subtle statistical correlations as it no longer needs to rediscover the differential equations underlying this numerical method.

4. *Improving prediction for longer lead times at the potential expense of short lead time predictions*

- this could be done by encouraging the model to avoid learning the chaotic and unpredictable dynamics by introducing noise into the features. The principle is the same as in variational autoencoders, and the model should learn that similar states should lead to similar predictions. This is of course untrue in real life, and small perturbations can lead to dramatic changes. But such a model would possibly learn better what is predictable in the more distant future with greater accuracy by making less definite predictions in the short term.

5. *Extracting a higher resolution data set* - the authors mentioned that GraphCast is better viewed as a family of models, where future iterations of it may enjoy a greater resolution. The limiting factor for its low resolution now according to the paper is the lack of quality high-resolution data. Perhaps it is possible to extract higher resolution data from the ERA5 dataset by interpolating temporally with the help of numerical weather prediction methods and spatially by looking at the neighboring grid cells.

As mentioned before, I do not find any of these extensions novel or propitious enough, and I am open to a different project. However, of these the second extensions is the most likely to be of value, so I will further elaborate on it.

GraphCast works by only looking at two most recent time points to make its prediction. This would be sufficient if the Markov assumption held true, but given the coarseness of the encoding of the weather system as well as all of the latent variables not in the dataset, it is easy to see why it does not hold. Therefore, introducing additional information about how the system looked and behaved further into the past, could allow the model to extract more temporally lengthy patterns. Of course with the addition of more states, the complexity of the model outpaces the improvements. As the authors state, even just adding a third state X^{t-2} is already more trouble than it is worth. I suggest then to add such a third state but to have it be a summary of n previous states.

This summary could be extracted by simple statistical measures such as the mean and the standard deviation. But these may be too simple and fail to capture useful relations. Instead, I suggest a transformer network to be prepended to the GraphCast architecture which extracts this context state automatically. It can be trained then in tandem with GraphCast to learn exactly the type of information that is useful to have.

This approach for extending GraphCast has one big issue in a project like this however. The model would need to be retrained, which is not something I will have the resources for in all likelihood. Therefore, I do know how realistic it is in the scope of this course.