`XTBTSScreener.jl` - Saving CPU Cycles with Julia and Machine Learning

Jackson Warner Burns *Computational Science and Engineering, MIT*

April 2023

Abstract

In chemical kinetics proposed reaction transition states are optimized with computationally expensive simulations for subsequent analysis and investigation. Often these simulations will fail to converge and require multiple iterations to achieve the desired result which wastes compute hours and impedes research. Using Julia and Machine Learning `XTBTSScreener.jl` is created as a proof-of-concept to predict which proposed transition states are likely to converge *before* investing CPU cycles for expensive simulations. `XTBTSScreener.jl` uses the output from inexpensive semi-empirical simulations, which can be executed in only a few minutes on consumer-grade hardware, to predict if the partially-optimized structure is likely to converge. The results of this initial investigation are promising and indicate that a combination of the atomic coordinates and higher-order electronic descriptors can enable a Long Short-Term Memory RNN (Hochreiter and Schmidhuber 1997) to achieve better-than-baseline performance on this task.

Background

In chemical kinetics, quantum mechanics simulations are often used to probe the complexities of novel reactions. One particular use case is the calculation of rate constants (Spiekermann, Pattanaik, and Green 2022). This requires predicting the three-dimensional electronic structure of the starting materials, the products, and - most importantly - the transition state between the two. The former two requirements are relatively simple and existing computational chemistry packages readily perform this task. Finding transition states is substantially more difficult. The typical workflow for doing so is shown in Figure 1 and explained in-depth below.

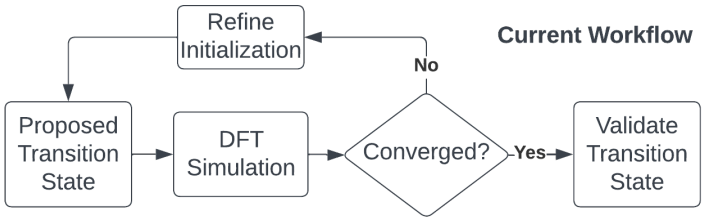

Figure 1: Current Transition State Search Workflow

First a 'guess' at the possible transition state is created. This initialization has historically been built manually with expert input and computationally inexpensive, but less accurate, simulation methods. One class of such methods which are used in this investigation are the "semi-empirical" methods which use shortcuts in solving the Hamiltonian based on experimental observations and human intuition. Systems are now being developed to predict transition states directly (Makoś et al. 2021) but they are imperfect. Next the possible transition state is subjected to an expensive quantum mechanics simulation. In this study, Density Functional Theory (DFT) is used to optimize the *proposed* transition state to hopefully arrive at a *valid* transition state.

Validity is determined by subsequent steps in the workflow which are beyond the scope of this study, as they involve applying further quantum mechanics simulations. This work

instead focuses on predicting if the DFT simulation will *finish execution normally* (i.e. the simulation "converged") rather than terminating mid-execution, which is a risk inherent in the process of proposing transition states. Termination has a variety of different and hard-to-detect causes. If a proposed transition state is trapped in a local minimum on the energy surface or on an optimization trajectory that is not productive, DFT simulations will fail to converge.

This challenge in and of itself would not be worth addressing if it were not for the cost of failed simulations. DFT scales by $O(n^3)$ for n electrons in the system, thus simulation times can be weeks or longer for common systems of interest. If not converged, this entire process must be repeated until the DFT simulation converges. Every failed calculation effectively wastes hundreds of compute hours and dramatically slows research progress.

To accelerate this workflow it would be useful to estimate *a-priori* if a proposed transition state is *"likely to converge"* before simulating with DFT. By generating a dataset of proposed transition states and their failure or convergence label, it is possible to train a machine learning model that is capable of making said prediction: `XTBTSScreener.jl`. This is shown schematically in Figure 2.

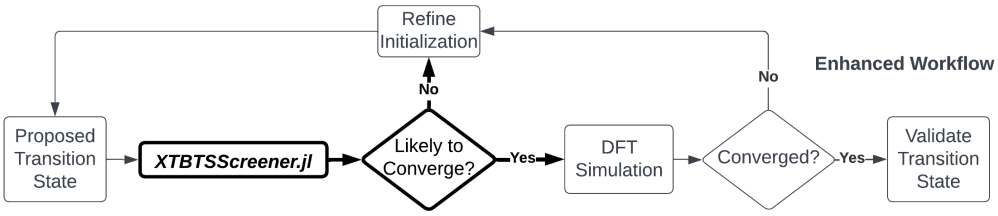

Figure 2: Proposed Enhanced Transition State Search Workflow

This diagram adds a new step before proceeding to DFT simulation in which proposed transition states that are *immediately* thought to be unlikely to converge will be rejected. The method by which the proposed transition state was generated would then be responsible

for creating a new, better-informed guess before continuing in the loop. In doing so, failed DFT simulations are avoided and the loop operates faster. Owing to the speed of Julia this screening step can be performed quickly so as not to defeat the purpose of rapid transition state proposal and simulation.

Data Preparation

Members of the William H. Green Group in MIT Chemical Engineering have previously generated a dataset containing many thousands of expert-suggested proposed transition states for reactions of interest in the chemical kinetics field. These structures were partially optimized using Extended Tight Binding semi-empirical quantum mechanics simulations (xTB), which is a computationally inexpensive method to arrive at a reasonable initialization (Bannwarth et al. 2021). All examples were then carried forward to DFT simulation, resulting in both converged and failed simulations. Currently, the data resides in log files output from the quantum chemistry simulation software that ran the simulations. Data were retrieved synchronously with this study but the precise implementation details are beyond the scope of investigation.

Approximately 20,000 simulation results were parsed and exported into an SQL database and then a CSV file for easy loading into Julia. Initial investigation revealed that the data were unbalanced (approximately 80% converged), so for this proof of concept the data were further downsampled to 7,000 samples with an even mix of failed and converged simulations. At the scale of the entire dataset with *millions* of samples, the balance between failed and converged simulations is naturally much closer.

The data actually retrieved from the log files were the standardized atomic coordinates of the proposed transition state structures and the Gibbs free energy, $E_0 + ZPE$, and number of simulation steps for each of the three sub-steps in the xTB simulation. Atomic coordinates reflect the actual orientation of the atoms in space and embed chemical data like bond length and angles. They are standardized by the simulation software such that

the center of mass is also the center of the simulation box. The Gibbs free energy and $E_0 + ZPE$ (sum of electronic total energy and zero point energy) are two higher-order electronic descriptors that reflect the relative energy of the transition state to its ground state. Number of simulation steps is included as a generic metadata descriptor for the xTB simulations describing at a high level the 'difficulty' of the simulation. Finally, the label for each point is simply whether or not the corresponding DFT simulation converged.

Methods

`XTBTSScreener.jl` is implemented using Julia v1.8.5, and all source code and results are publicly available on GitHub (github.com/JacksonBurns/xtb-ts-screener). The grander vision for `XTBTSScreener.jl` and its derivatives would be within a much larger closed-loop optimization tool, so the speed of Julia will be critical in making this approach worthwhile. `Lux.jl` (Pal 2022), a low-code explicit parameterization-driven neural network package, is used to implement the model. Sixty input dimensions and six hidden dimensions with sigmoid activation were used alongside the binary cross-entropy function to quantify loss, which is limited in numerical stability but offers benefits in interpretability.

The model itself is a Long Short-Term Memory recurrent neural network (Hochreiter and Schmidhuber 1997), which is able to process higher-dimensional inputs like those in this study. This modeling approach was chosen for its flexibility and ease of extensibility in future efforts. During the proposal of the transition state a sequence of atomic coordinates are generated, and it may be possible to assemble these into a 'movie' on which the RNN can be trained. The ubiquitous Adam optimizer (Kingma and Ba 2017) is used in model training to enable more rapid convergence and because of its suitability for large datasets. `Zygote.jl` (Innes 2018) is also used to provide automatic differentiation capabilities for the network.

To parameterize the transition states the aforementioned parameters are concatenated into a single input array. Owing to the nature of chemical systems, the transition states in this

dataset range in size from thirty to fifty atoms, with many skewing to the larger side. To maintain a uniform encoding and preserve interpretability of feature arrays, zero-padding is used for transition states with fewer than fifty atoms. As later explained in the Future Work section, this simple encoding strategy should likely be replaced in the future with a more robust approach but as a proof of concept it is sufficient.

Results

There is no established best practice for training an LSTM on this type of data so parameters like the batch size, learning rate, and number of epochs were determined by trial and error. To establish a baseline model performance against which subsequent iterations could be compared, a network was trained using an off-the-shelf configuration. Only the atomic coordinates are provided as features, the data were *not* downsampled, and the NN hyperparameters were left to their defaults as specified in the `Lux.jl` tutorial (Pal 2022): learning rate of 0.01, batch size of 128, and 25 epochs. This produced the loss and accuracy plot show below.

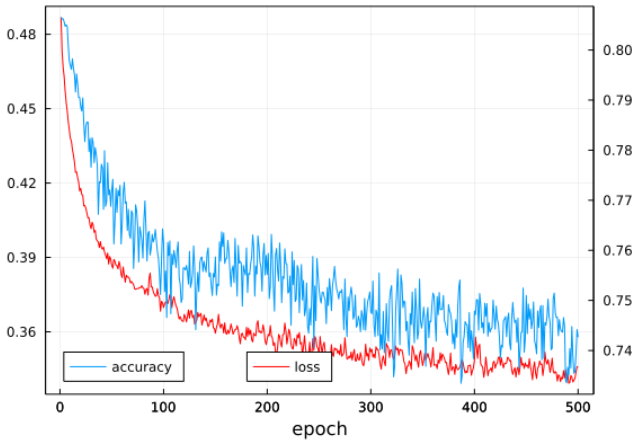

Figure 3: Baseline Modeling Results

As epochs increase the loss continually decreases though in a noisy manner, which is

not entirely unexpected for the binary cross-entropy loss function. Accuracy however decreases across epochs despite this decrease in the loss function, which is a product of this uninformative parameterization scheme. When the model is randomly initialized it achieves an accuracy of approximately 80% This seems impressive but is actually the converse of the failure rate for the non-downsampled dataset of 20,000 transition states. These facts indicate that the network hyperparameters were not conducive to learning and/or the embedding is masking information. Subsequent iterations on the model start by addressing the latter.

During the initial transition state proposal step, molecular descriptors other than the atomic coordinates are generated: Gibbs free energy, $E_0 + ZPE$, and number of simulation steps (as discussed in Methods). These can be prepended to the atomic coordinates to arrive at an 'augmented' feature array. The results of training on such a representation are shown below.

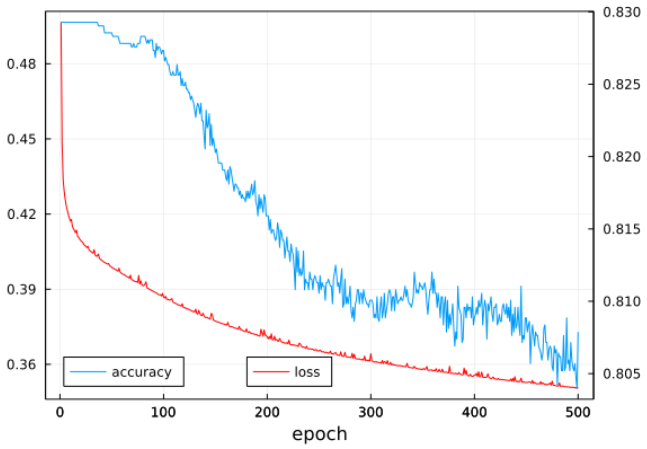

Figure 4: Augmented Features Modeling Results

With these additional descriptors the model performance stays almost flat across the epochs, losing only a few percentage points of accuracy from the initialization. The loss curve also smooths significantly without changing the loss function. These changes indicate that the new embedding is an improvement but that the first issue with the baseline model

must now be addressed: the hyperparameters.

Though trial and error, an ideal configuration for the network was found. Initial learning rate was varied from 0.01 to 0.0001, the latter of which was empirically determined to be critical for model convergence. Batch sizes from 16 to 128 samples were used with 64 found to be the ideal value to avoid over-fitting while still allowing convergence. Number of epochs was allowed to go as high as many thousands and as low as only ten, but with the above combination or parameters the loss function leveled out significantly after only fifty epochs. Finally, the data were downsampled to reach an even balance of converged and failed samples. This provides a more meaningful way to evaluate model performance since the baseline of random guessing would yield a result of exactly 50% and the network can move an equal amount in either a positive or negative manner. The accuracy and loss curve for this final configuration are shown below.

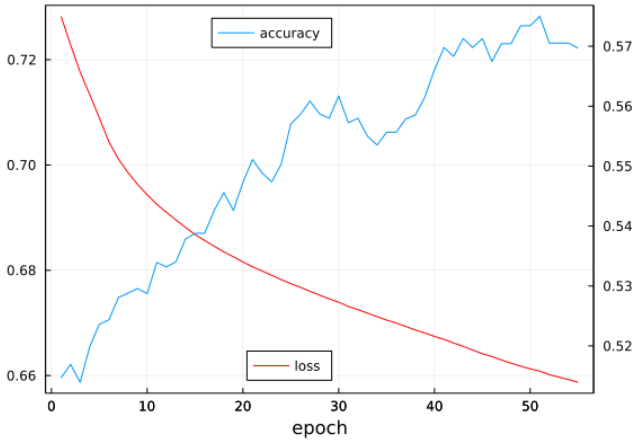

Figure 5: Final Modeling Results

The loss curve was completely stabilized, and although the accuracy was still somewhat volatile it showed a general increase of approximately 7% over the baseline performance after only 50 epochs. While not up to production standard this model is effectively a proof of concept for the use of LSTM with augmented embeddings.

Conclusions

Using Julia and the `Lux.jl` package a Long Short-Term Memory Recurrent Neural Network was trained on a set of proposed reaction transition states to predict if said transition state is likely to converge in subsequent quantum mechanics simulations. The embedding for the transition states was derived from easily retrieved descriptors and promises excellent scalability. The model achieved a 7% increased performance over baseline, which is not itself production ready but is evidence that this computationally inexpensive approach to screening could be added to the current workflow to save CPU cycles and accelerate discovery.

Future Work

First and foremost the dataset should be expanded with a focus on negative examples to alleviate the imbalance issue. The embedding should also be further augmented with information that can be retrieved from the transition state proposal step such as the absolute energies or vibrational frequencies. Such improvements would still require zero padding, and although it is one of the simplest approaches the possibility of changing to an autoencoding scheme instead for increased performance and problem specificity should be investigated. Alternative architectures should also be evaluated, particularly the Graph Neural Network (GNN) such as those in `GraphNeuralNetworks.jl` (Lucibello and contributors 2021). Literature precedent from the chemical informatics field at large indicates that GNNs often perform better than typical NNs on chemical data, and improvements made in these fields may also be applicable here.

Acknowledgements

The author acknowledges Green Group member Haoyang Wu for performing the calculations and providing the data which were used in this study and the MIT SuperCloud and

Lincoln Laboratory Supercomputing Center for providing HPC resources that contributed to the generation of this dataset and training of `XTBTSScreener.jl` (Reuther et al. 2018).

References

Bannwarth, Christoph, Eike Caldeweyher, Sebastian Ehlert, Andreas Hansen, Philipp Pracht, Jakob Seibert, Sebastian Spicher, and Stefan Grimme. 2021. "Extended Tight-Binding Quantum Chemistry Methods." *WIREs Computational Molecular Science* 11 (2): e1493. https://doi.org/https://doi.org/10.1002/wcms.1493.

Hochreiter, Sepp, and Jürgen Schmidhuber. 1997. "Long Short-Term Memory." *Neural Computation* 9 (8): 1735–80. https://doi.org/10.1162/neco.1997.9.8.1735.

Innes, Michael. 2018. "Don't Unroll Adjoint: Differentiating Ssa-Form Programs." *CoRR* abs/1810.07951. http://arxiv.org/abs/1810.07951.

Kingma, Diederik P., and Jimmy Ba. 2017. "Adam: A Method for Stochastic Optimization." http://arxiv.org/abs/1412.6980.

Lucibello, Carlo, and other contributors. 2021. "GraphNeuralNetworks.jl: A Geometric Deep Learning Library for the Julia Programming Language." https://github.com/CarloLucibello/GraphNeuralNetworks.jl.

Makoś, Małgorzata Z., Niraj Verma, Eric C. Larson, Marek Freindorf, and Elfi Kraka. 2021. "Generative Adversarial Networks for Transition State Geometry Prediction." *The Journal of Chemical Physics* 155 (2): 024116. https://doi.org/10.1063/5.0055094.

Pal, Avik. 2022. "Lux: Explicit Parameterization of Deep Neural Networks in Julia." *GitHub Repository*. https://github.com/avik-pal/Lux.jl/; GitHub.

Reuther, Albert, Jeremy Kepner, Chansup Byun, Siddharth Samsi, William Arcand, David Bestor, Bill Bergeron, et al. 2018. "Interactive Supercomputing on 40,000 Cores for Machine Learning and Data Analysis." In *2018 Ieee High Performance Extreme Computing Conference (Hpec)*, 1–6. IEEE.

Spiekermann, Kevin, Lagnajit Pattanaik, and William H. Green. 2022. "High Accuracy Barrier Heights, Enthalpies, and Rate Coefficients for Chemical Reactions." *Scientific Data* 9 (1): 417. https://doi.org/10.1038/s41597-022-01529-6.