Predicting Chemical Reaction Type and Reaction Products with Recurrent Neural Networks

Anita Gupta (avgupta@stanford.edu)

Abstract
• Synthesis of chemical molecules is crucial in medicine, environmental science, materials, and more disciplines. However, predicting the outcomes of synthesis reactions is low accuracy, & mistakes are costly and time consuming.
• Here, we use deep recurrent and convolutional neural networks to 1. Classify the type of a chemical reaction given the reactants.
  2. Generate the correct products given the reactants.
• Reaction prediction classifier achieves >90% accuracy and AUPRC.
• Generated products have high degree of chemical similarity to actual products, and we outperform the oracle (rule based prediction).

Data
• 50,000 Curated Reactions from US Patents - across 50 Reaction Types. Reactions represented as strings using the SMILES grammar [1]
Data Preprocessing and Featureization
• Removal of Atom Mapping, Canonicalize & Sort Reactants
• Preprocessing by Length - Middle 80% of Data was retained by length, no reactions where (react) < 30 char.
• Removal of Salts - Left over from synthesis conditions (to stabilize molecule in some state); unnecessary for core reaction
• Tokenization: 54 total tokens, each an element or number. “O” and “E” are sentinels for “start” and “end”, respectively.

Models: Reaction Type Classification
• Baseline: Logistic Regression (Weighted for Class Imbalance)
• Features: Fingerprint, bitstring with presence of substructures
• Recurrent Neural Network Classifier
• Learn features directly from SMILES string!
• Two LSTM Layers with Hidden State = 512 x 1
• Batch Norm, Dropout = 0.3 (Layer 1), 0.5 (Layer 2)
• Output Dense Layer for probability of 50 types
• Data split 60% Train, 20% Validation, 20% Test Set

Models: Reaction Product Generation
• Baseline: Ngrams model with n=5, frequencies computed per reaction type
• "Oracle": Rule-Based Prediction - Extraction of reaction center
Vanilla Seq2Seq Architecture
• Encoder: reactants input, latent state output
• Decoder: takes in latent state and reproduces products
• Both encoder and decoder have 2 GRU Layers
• Hidden state size 500 on which decoder is conditioned

Seq2Seq with Attention (RNN Encoder/Decoder)
• Encoder - Bidirectional GRU, 2 Layers, Dropout = 0.05, Hidden State=500
• Decoder - 2 GRU Layers, Hidden State=500
• Attention: At each step, decoder outputs "query"
• Attn Weights = softmax("query" x encoder_outputs)
• Rnn type optionally added as special token before input

Discussion
• Reaction Type Classifier (Dropout 0.1/0.1) has >95% accuracy and AUPRC of 0.91, significantly outperforms baseline
• Diagnosed overfitting and improved generalization by varying layers of dropout.
• Error Analysis - Sought to understand performance as function of class/imbalance. AUPRC above 0.9 for all classes (Fig 10), not significantly worse for imbalanced classes (Fig 9).

Conclusion
• Seq2Seq with Attention (RNN Encoder/Decoder) is the best performing of the three models designed; Tanimoto similarity averages 0.86 and reaches at 0.95, higher than “rule based” system (0.80 tanimoto).
• Why? Rule-based extraction needs correct extraction of reaction centers (place where reaction is taking place); requires a great deal of domain knowledge, libraries implementing this (rdkit) are buggy.
• Figure 5 shows that even on incorrect outputs, the predicted output shows motifs from the actual output
• Inaccuracy is higher on larger output molecules (as expected)
• Seq2Seq (CNN Encoder) surprisingly did not reach the same level of performance (or outperform Vanilla Seq2Seq).

References

I would like to acknowledge Microsoft for the Azure compute credit.