

平成 27 年度 TSUBAME 産業利用トライアルユース 成果報告書

利用課題名 大規模分散深層学習の創薬への応用

英文: Application of Massive Distributed Deep Learning toward Drug Discovery

岡野原 大輔

Daisuke Okanohara

株式会社 Preferred Networks

Preferred Networks, Inc.

<https://www.preferred-networks.jp/>

本プロジェクトでは、深層学習による表現学習の創薬への応用として、化合物の Finger Print を学習ベースで獲得する Neural Finger Print (NFP) を再現し、さらにその乱択アルゴリズム化である Drop Edge を提案した。さらに、両者の化合物活性予測のタスクを TSUBAME 上で実施し、予測精度と速度を比較した。72 アッセイの化合物活性予測に対して NFP と Drop Edge を適用し、特に半径(アルゴリズム中のイテレーション回数)が大きい場合に、1 割程度の計算時間削減を確認した。一方で、予測精度は Drop Edge の方が 1 割程度低い(Accuracy で比較)という結果も得られた。計算量と予測精度のトレードオフの解消は今後の課題である。

This project explores an application of representation learning by deep learning toward drug discovery. Specifically, we reproduce Neural Finger Print (NFP), which learns the finger print of compounds from data, and propose the randomized algorithm of NFP, called Drop Edge. We apply these two methods to the activity prediction task of 72 assays and find that Drop Edge can reduce the time complexity, especially when the radius is large, by approximately 10%. On the other hand, prediction accuracy is also reduced by approximately 10%. So there remains the trade-off between time complexity and accuracy.

Keywords:

Neural Finger Print、機械学習、深層学習、QSAR

背景と目的

定量的構造活性相関 (Quantitative Structure-Activity, QSAR) は化合物の化学的な性質と毒性や体内の酵素に対する代謝の度合いなど、生化学的性質間の量的な関連を指す。QSAR は創薬での候補化合物探索において、初期のスクリーニングに利用される。精度の良い QSAR は、スクリーニング精度向上に貢献し、創薬コストの削減、化合物のヒット率上昇につながる。

化合物の探索空間全体は 10^{60} 程度と言われており [Bohacek+96]、これらの活性をすべて実験するのは現実的ではない。そこで探索の初期では、計算機上で化合物ライブラリ中の候補化合物の QSAR を予測する Virtual Screening の手法を取り入れるのが一般的である。

化合物と受容体の結合の様子をシミュレーションするドッキングシミュレーションを初めとして、様々な Virtual Screening の手法が開発されている。その中

でも特に深層学習による Virtual Screening の手法が近年注目を集めている。深層学習は層が深いニューラルネットワークを用いた機械学習の手法の総称であり、動画像解析(検出・認識)・自然言語処理(翻訳・対話)など様々な分野のタスクで既存の手法を超える予測精度が報告されている。

これまでに、深層学習を用いた化合物の活性予測として [Dahl+14], [Thomas+14], [Divenaud+15] などの研究がある。[Dahl+14] の手法は、化合物の構造式を Finger Print と呼ばれる固定長のビットベクトルに変換し、Finger Print から特定のターゲットに対する活性の有無を予測する 2 値分類問題を、ニューラルネットワークを用いて解いている。さらに、複数のターゲットに対する活性予測を共通のニューラルネットワークで予測する、マルチタスク学習を行うことで予測精度が向上することを報告した。同様のマルチタスク深層学習による活性予測は [Thomas+14] などでも行われている。

いずれも研究でも化合物の構造式から Finger Print

への変換方法には大きな注目をしていなかった。[Dahl+14]が利用している商用ソフトウェア DRAGON は、Finger Print のビットごとに、そのビットを立てる条件をルールベースで記述している。また、[Thomas+14]では、標準的な Finger Print の構築アルゴリズムの Extended-connectivity fingerprint (ECFP)を利用している。いずれも、なぜその Finger Print を利用するかについては特に言及はしていなかった。

今日の深層学習の成功の理由の1つとして、表現学習、すなわち、学習ベースによる入力データの特徴抽出を高精度に行えることが挙げられることが多い。構造式から Finger Print への変換を特徴抽出と捉えれば、Finger Print を学習ベースで獲得するのは自然な発想だと言える。[Divenaud+15]では化合物を複数種類のノードから構成されるグラフとして捉え、グラフ上の畳み込み演算として Finger Print を獲得する手法、Neural Finger Print (NFP)を提案した。

概要

本プロジェクトではまず NFP の再現を行った。我々の再現では、NFP は非常に計算コストが高く、そのままの手法で大規模化するのには難しいと考えた。そこで、本プロジェクトの2つ目の取り組みとして、NFP の手法を乱択アルゴリズム化し、計算量を落とす手法 Drop Edge を提案し、NFP との計算量・精度の比較を行った。

手法

まず、ECFP, NFP, Drop Edge の手法を解説する。

ECFP は、化合物を長さ d の固定長のビットベクトルに変換する決定的なアルゴリズムである。予め可変長の引数を取り 1 つの整数値を返すハッシュ関数を用意して固定する。まず、各原子に初期ハッシュ値を割りあてる。この値は原子の種類が同一ならば同一の値を割り振る。また、出力のビットベクトルの全ビットを 0 に初期化する。次に、分子中のそれぞれの原子に対し、その原子自身とその原子に隣接する原子達に割り当てられたハッシュ値を集め、ハッシュ関数に適用する。原子に割り振られたハッシュ値をその出力値で更新する。ま

た、ここで得られた出力値を d で割った剰余を計算し、ビットベクトルの対応するビットを 1 に立てる。分子中の全原子に対してこのハッシュ値の更新を同時に行う過程を 1 イテレーションとし、これを決められたイテレーション数 R 回行う。 R 回のイテレーションを行った後に得られた後のビットベクトルをこの化合物の Finger Print とする。 R 回のイテレーションを行うと、ある原子から R だけ離れた原子の情報まで含めて Finger Print を作成するので、イテレーション数 R を半径 (radius) と呼ぶ。

NFP の構築アルゴリズムは ECFP のそれと共通部分が多いが以下の点が異なる。図 1 は ECFP と NFP の手法の擬似アルゴリズムの比較である。

- ECFP では各原子にはハッシュ値を割り振るのに対し、NFP では長さ f の実数値ベクトルを用いる。
- ECFP では各原子の値の更新にハッシュ関数を用いるのに対し、NFP では 1 層のニューラルネットを用いる。
- ECFP で原子と隣接する原子達のハッシュ値を連結してハッシュ関数に適用した代わりに、すべてのベクトルの和を取りニューラルネットに入力する。
- Finger Print に値を追加する際、ECFP では剰余に対応する特定の 1 ビットが 1 で残りが 0 という、ハードな割り当てを行う。それに対し、NFP では 1 層のニューラルネットを用いて確率分布 (すなわち各要素は 0 以上で合計が 1 の出力) によるソフトな割り当てを行う。

Algorithm 1 Circular fingerprints	Algorithm 2 Neural graph fingerprints
1: Input: molecule, radius R , fingerprint length S	1: Input: molecule, radius R , hidden weights H_1^1, \dots, H_R^R , output weights W_1, \dots, W_R
2: Initialize: fingerprint vector $f \leftarrow 0_S$	2: Initialize: fingerprint vector $f \leftarrow 0_S$
3: for each atom a in molecule	3: for each atom a in molecule
4: $r_a \leftarrow g(a)$ \triangleright lookup atom features	4: $r_a \leftarrow g(a)$ \triangleright lookup atom features
5: for $L = 1$ to R \triangleright for each layer	5: for $L = 1$ to R \triangleright for each layer
6: for each atom a in molecule	6: for each atom a in molecule
7: $r_1 \dots r_N = \text{neighbors}(a)$	7: $r_1 \dots r_N = \text{neighbors}(a)$
8: $v \leftarrow [r_a, r_1, \dots, r_N]$ \triangleright concatenate	8: $v \leftarrow r_a + \sum_{i=1}^N r_i$ \triangleright sum
9: $r_a \leftarrow \text{hash}(v)$ \triangleright hash function	9: $r_a \leftarrow \sigma(v H_L^L)$ \triangleright smooth function
10: $i \leftarrow \text{mod}(r_a, S)$ \triangleright convert to index	10: $i \leftarrow \text{softmax}(r_a W_L)$ \triangleright sparsify
11: $f_i \leftarrow 1$ \triangleright Write 1 at index	11: $f \leftarrow f + i$ \triangleright add to fingerprint
12: Return: binary vector f	12: Return: real-valued vector f

図 1: ECFP (左) と NFP (右) の比較 ([Divenaud+15] より引用)

化合物の最大次数を N とする。NFP では、予め、長さ f のベクトルを長さ d のベクトルに変換するニューラルネット H と、各次数 n ($n=1, \dots, N$) に対し、長さ f のベクトルを同じ長さ f のベクトルに変換するニューラルネット W_n

を用意する。H は各要素が 0 以上で要素の合計が 1 になるように正規化され、出力が確率として解釈できるようにする。典型的には、 W_n は 1 層の全結合層 + 活性化関数、H は 1 層の全結合層 + Softmax 関数で実現される。

具体的な構築アルゴリズムは以下の通りである。まず、各原子を原子の種類ごとに長さ f の実数値ベクトルで初期化する。また、Finger Print を長さ d の 0 ベクトルで初期化する。次に、分子中のそれぞれの原子に対し、その原子自身とその原子に隣接する原子達に割り当てられたベクトルの和をとり、 W_n にフォワードする。ここで、 n はこの原子の隣接する原子の数(次数)である。また、同時にこの和を H にもフォワードし、出力値を Finger Print に足す。分子中の全原子に対してこのハッシュ値の更新を同時に行う過程を 1 イテレーションとし、これを決められたイテレーション数 R 回行う。

図 2 は特定の 1 原子に割り当てられたベクトル(v_{11})の更新とアルゴリズムの出力である Finger Print(v)の更新を図示したものである。

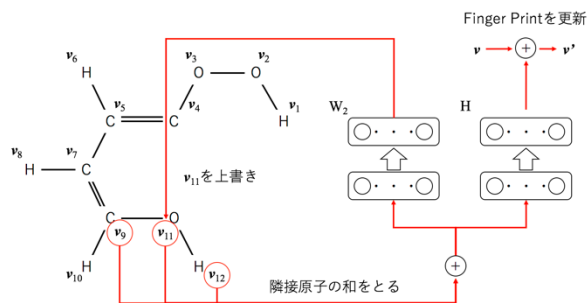


図 2: NFP でのベクトル更新

NFP では、最終的な出力である Finger Print が、各原子に割りあてる初期値ベクトル、重みの更新に利用するニューラルネット W_n のパラメータ、および Finger Print への出力に用いるニューラルネット H のパラメータに関して微分可能な関数になっている。そのため、これらの値は誤差逆伝播により学習可能である。

最後に、NFP の乱択化である Drop Edge を解説する。NFP では、ある原子に割り振られた長さ f の実数値ベクトルを更新する際、その原子に隣接するすべての原子のベクトルを用いた。Drop Edge では、隣接する全ての原子を用いる代わりに、ランダムに 1 個の原子を

選びそれのみを利用する。化合物をグラフと見た時の最大次数を N とすると、NFP では、 $N+1$ 種類のニューラルネット(W_1, \dots, W_N, H)を用意する必要があった。それに対して、Drop Edge で用意すべきニューラルネットは 2 つのみである。これにより学習すべきパラメータ及びメモリ使用量が減少する。また、NFP に比べ行列計算の回数が減るので、所要時間が削減されることが期待出来る。もちろん、乱択化により訓練時に利用する情報は減少するため、予測モデルの精度が落ちる可能性がある。実験では乱択化による精度劣化の度合いを検証した。

実験

72 アッセイに対して、化合物の活性の有無を予測する 2 値分類問題を NFP と Drop Edge を用いて解き、その精度と実行速度を比較した。

モデルは、Finger Print の生成する生成器と、生成した Finger Print を用いて活性の有無を予測する分類器の 2 つからなる。生成器として NFP または Drop Edge を用い、分類器はニューラルネットを用いてモデル化している。

学習は、生成器を学習する Training ステップと、生成した Finger Print を評価する Validation ステップの 2 つに分かれている。それぞれのステップに対し、Training データセット 3500 個と、Test データセット 500 個の化合物を用意する(すなわち 1 アッセイあたり合計 $(3500 + 500) \times 2 = 8000$ 個の化合物を利用する)。Training データセットは分類器を学習に、Test データセットは分類精度を評価に用いる。それぞれのデータセットには活性ありとなしが半数ずつ含まれるようにした。

生成器は Training ステップのみで更新を行い、Validation ステップでは生成器を学習させないようにする。分類器はすべてのデータセットに関して更新を行う。前の Epoch での Validation ステップでの分類器の学習を通じて、Validation データセットでの情報が生成器の方に次の Epoch で伝わってしまうため、本来は分類器の重みは Training ステップと Validation ステップの間で初期化すべきであるが、今回は実験時間の都合上、その実験は実施しなかった。初期化の有無によ

る精度変化は今後の検討事項の 1 つである。

本実験に利用したコードを以下の URL で公開している。レポジトリの nfp ディレクトリに NFP の、drop_edge ディレクトリに Drop Edge を実装している。

URL: <https://github.com/pfnet/nfp/>

結果および考察

実験は以下のようなパラメータで行った。後述の結果の通り、この実験設定で、1 つのアッセイの実験に 10-20 時間程度要する。

- Finger Print 長 d : 128
- 特徴ベクトルの長さ f : 128
- 半径 R : 3 または 6
- 分類器ニューラルネットの中間層のノード数: 256
- ミニバッチサイズ: 50
- Training ステップでの Training データセットによる学習のエポック数: 15
- Validation ステップでの Training データセットによる学習のエポック数: 20
- 全体のエポック数: 15

図 3 は 72 アッセイに対する Validation ステップの Test データセットに対する Accuracy の 15 エポックのうちの最大値をプロットしたものである。base は従来手法、drop は Drop Edge であり、R=6 と書かれているものは半径 R を 6 に設定したもの、書かれていないものは半径 R を 3 に設定したものである。Drop Edge の精度は従来手法のおよそ 0.9 倍に落ちている。半径 R を増やすことによって、より遠い距離の隣接関係に関する情報が NFP に含まれることになり、精度が向上することを予想していたが、精度に大きな変化は見られなかった。

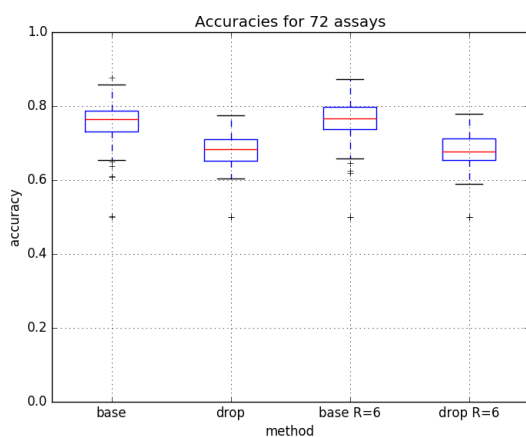


図 3: Accuracy の最大値の box plot

図 4 は 72 アッセイに対する計算時間をプロットしたものである。本手法の計算量は半径 R に比例するものであり、実際の所要時間でも $R=6$ の所要時間は、 $R=3$ の倍程度時間がかかっている。NFP と Drop Edge の所要時間を比較すると、 $R=3$ では大きな改善は見られなかったが、 $R=6$ では、1 割程度の所要時間削減がなされていることがわかる。

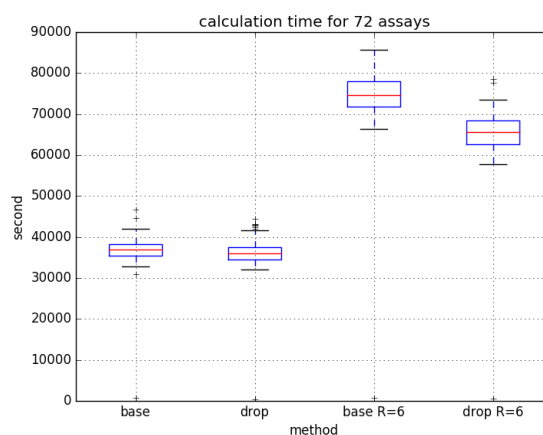


図 4: 実行時間の box plot (横軸の単位は秒)

まとめ、今後の課題

本プロジェクトでは、深層学習による表現学習の創薬への応用として、化合物の Finger Print を学習ベースで獲得する Neural Finger Print (NFP) を再現し、さらにその乱択アルゴリズム化である Drop Edge を提案した。さらに、両者の化合物活性予測のタスクを TSUBAME で実験し、その予測精度と速度を比較した。

72 アッセイの化合物活性予測に対して NFP と Drop Edge を適用し、特に半径 (アルゴリズム中のイテレーションの回数) が大きい場合に、計算時間を 1 割程度削減することができることを確認した。一方で予測精度は Accuracy で測定した時に 1 割程度 Drop Edge の方が低いという結果も得られた。本手法では、計算時間と予測精度のトレードオフで、計算時間の有利な選択肢を提案することはできたが、トレードオフ自体を緩和することはできなかった。

今後の課題として 3 つの可能性が考えられる。1 つ目は上記のトレードオフを解消する乱択化手法の開発である。2 つ目は、Drop Edge のメモリ使用量削減の効果検証である。Drop Edge は学習パラメータを削減し

ており、メモリ使用量の削減も見込まれる。今回のプロジェクトでは、時間の都合上この効果を実証検証することはできなかった。3 つ目はマルチ GPU、マルチノードによる高速化である。本プロジェクト期間中にもマルチ GPU による NFP, Drop Edge の並列化を試みたが、大きな改善は得られなかった。並列化のためにはさらなるアルゴリズムの改良が必要であると考えられる。

参考文献

- [Bohacek+96] The art and practice of structure-based drug design: a molecular modeling perspective. *Med. Res. Rev.* 16: 3–50.
- [Dahl+14] Dahl, G. E., Jaitly, N., & Salakhutdinov, R. (2014). Multi-task neural networks for QSAR predictions. arXiv preprint arXiv:1406.1231.
- [Duvenaud+15] Duvenaud, D. K., Maclaurin, D., Iparraguirre, J., Bombarell, R., Hirzel, T., Aspuru-Guzik, A., & Adams, R. P. (2015). Convolutional Networks on Graphs for Learning Molecular Finger Prints. In *Advances in Neural Information Processing Systems* (pp. 2215-2223).
- [Thomas+14] Thomas Unterthiner, Andreas Mayr, Günter Klambauer, Marvin Steijaert, Jörg Kurt Wegner, Hugo Ceulemans, Sepp Hochreiter. Deep Learning as an Opportunity in Virtual Screening. *Deep Learning and Representation Learning Workshop: NIPS 2014*