The 33rd Annual Conference of the Japanese Society for Artificial Intelligence, 2019

ノイズを含む遷移データからの論理プログラム学習

Learning Logic Programs from Noisy State Transition Data

ポア インジュン *1 井上 克巳 *1*2
Yin Jun Phua Katsumi Inoue

*1東京工業大学 *2国立情報学研究所
Tokyo Institute of Technology National Institute of Informatics

Real world data are often noisy and fuzzy. Most traditional logical machine learning methods require the data to be first discretized or pre-processed before being able to produce useful output. Such short-coming often limits their application to real world data. On the other hand, neural networks are generally known to be robust against noisy data. However, a fully trained neural network does not provide easily understandable rules that can be used to understand the underlying model. In this thesis, we propose a Differentiable Learning from Interpretation Transition (δ-LFIT) algorithm, that can simultaneously output logic programs fully explaining the state transitions, and also learn from data containing noise and error.

1. はじめに

解釈遷移からの学習 (Learning from Interpretation Transition; **LFIT**) はダイナミックなシステムから解釈可能なルールを学習するアルゴリズムである．ダイナミックなシステムから観測した状態遷移を入力とし，LFIT アルゴリズムはその状態遷移を満たす論理プログラム (normal logic program; **NLP**) を出力する．得られた NLP より，システム内の各コンポーネントの関係性が明らかになる．その関係性を利用し，システムの将来的な状態の予測や，ある理想の状態への制御が可能となる．ロボット工学の分野では，周りの環境をモデル化することによりプランニングなどが可能となる [5]．生物の分野では，遺伝子間の関係性が新しい治療の発見などにつながる [7].

LFIT アルゴリズムは主にシンボリック手法 [6] とニューラルネットワーク手法 [3] の二つの手法で実装されている．シンボリック手法では論理演算などを用いて NLP を学習する．入力の状態遷移はすべて正例および負例として捉え，各遷移ごとに学習した NLP がすべての例を満たすように論理演算を行う．これにより得られた NLP はすべての遷移に対して一貫であることが保証される．しかし，この手法にはいくつかの問題点があり，その一つとしてノイズが含んだ入力に対して頑健でない．

一方，統計的機械学習がノイズに対して頑健であるとされている [8, 9]．従って，ニューラルネットワーク手法ではノイズへの頑健性が期待されている．先行研究ではニューラルネットワークを用いて LFIT アルゴリズムを実装する可能性を示した [3]．しかし [3] では，ノイズへの頑健性が調べられていない．

一般的に統計的機械学習およびニューラルネットワークの機械学習手法が近年普遍化している．その理由の一つとして，実応用での有効性が普遍化を促していると考えられる．しかし多くの分野に応用されていくに連れ，いくつかの弱点が明らかとなった．学習を終えて得られたモデルが人間によって解釈不可能であることが医療分野などでの応用の大きな妨げとなっている．ニューラルネットワークは一般的にどの程度学習データ以外のものまで一般化できるか，なにを学習できたか調べるのが困難である．従って，データに隠れたバイアスが存在したときに，その問題を発見するのがとても困難である．

本研究では，微分可能な LFIT アルゴリズム (δ-LFIT) を提案する．δ-LFIT はエンドツーエンドで微分可能なアルゴリズムである．シンボリック手法でのノイズの頑健性のなさ，および統計的機械学習での解釈不能なモデルしか得られない問題を，δ-LFIT を通して解決する．δ-LFIT でのノイズに対する頑健性，さらに少量のデータから学習が可能であること，そして出力が人間にとって解釈可能であることを示す．

本論文ではまず，論理プログラムおよび LFIT に関する必要な知識について述べる．次に，提案手法の δ-LFIT のアルゴリズムを説明する．そして本研究で行われた実験をいくつか示し，最後に考察および関連研究について述べる．

2. 背景

ここでは本研究を理解する上で必要な論理プログラムおよび LFIT に関する背景知識について述べる．

2.1 論理プログラム

NLP は

$$A \leftarrow A_1 \wedge \cdots \wedge A_m \wedge \neg A_{m+1} \wedge \cdots \wedge \neg A_n \qquad (1)$$

の形式をとったルールの集合である．ここで A 及び A_i はアトムである ($n \geq m \geq 0$)．任意の (1) の形式をとるルール R において，A は R のヘッドであり $h(R)$ で表す．そして，$\leftarrow$ の右辺の連言は R のボディであり R のボディで現れるリテラルの集合を $b(R) = \{A_1, \ldots, A_m, \neg A_{m+1}, \ldots, \neg A_n\}$ で表し，R のボディで正リテラルとして現れるアトムを $b^+(R) = \{A_1, \ldots, A_m\}$，負リテラルを $b^-(R) = \{A_{m+1}, \ldots, A_n\}$ と表す．

NLP に出現する全てのアトムの集合をエルブラン基底 $\mathcal{B}$ と表す．エルブラン解釈 I はエルブラン基底 $\mathcal{B}$ の部分集合であり，各アトムへの真偽値の割り当てである．ルール R および解釈 I が与えられ，$b^+(R) \subseteq I$ および $b^-(R) \cap I = \emptyset$ が $h(R) \in I$ を含意する時，解釈 I はルール R を満足するという．無矛盾な解釈 I が P の全てのルールを満足する時，I は P のモデルである．論理プログラム P 及びエルブラン解釈 I を考える時，T_P オペレータは $T_P : 2^{\mathcal{B}} \rightarrow 2^{\mathcal{B}}$ の対応である．この時 T_P オペレータは

$$T_P(I) = \{h(R) \mid R \in P, b^+(R) \subseteq I, b^-(R) \cap I = \emptyset\} \qquad (2)$$

と定義する．

状態が同期的に変化していくシステムを表現するには，(1) を

$$A^{t+1} \leftarrow A_1^t \wedge \cdots \wedge A_m^t \wedge \neg A_{m+1}^t \wedge \cdots \wedge \neg A_n^t \qquad (3)$$

のようにダイナミックな形式にできる．ここで，t は時間ステップであり，$t+1$ は次の時間ステップとして考える．つまり，(3) では，時間 t で $A_1, \ldots, A_m$ が真，そして $A_{m+1}, \ldots, A_n$ が偽である時，時間 $t+1$ で A が真となることを表している．

2.2 LFIT

解釈遷移の集合 E が与えられた時，その全ての遷移 $(I, J) \in E$ について $J = T_P(I)$ を満たす論理プログラム P を学習するアルゴリズムを LFIT と呼ぶ [4]．いくつかの **LFIT** アルゴリズムが提案されている [3, 6]．LFIT の主な目的は対象システムのダイナミクスを説明する NLP を自動的に学習すること．

LFIT は，状態の遷移に注目している．解釈 I があるシステムの t 時点での状態を表している時，同システムの $t+1$ 時点での状態 J もまた解釈である．この場合，(I, J) が LFIT で注目している解釈遷移となり，システムが I から J へ遷移していることを表す．

ルール R および解釈遷移 (I, J) が与えられ，$b^+(R) \subseteq I$ and $b^-(R) \cap I = \emptyset$ が $h(R) \in J$ を含意する時かつその時に限り R は遷移 (I, J) と一貫であるという．R が得られた全ての解釈遷移 $I \in E$ と一貫してる時，R が E と一貫であるという．また，P のすべてのルール R が E と一貫である時，P が E と一貫であるという．

論理演算を用いた LFIT アルゴリズムの実装はいくつかある．ボトムアップ手法では最も特殊なルール，つまり最初はどんな解釈遷移とも一貫とならないようなルールから開始し，一般かを用いながら条件を弱めていく．トップダウン手法では最も一般なルール，つまり最初からどんな解釈遷移にも適応できてしまうようなルールから開始し，反例を見つけながら特殊化を繰り返し最小の NLP を学習する手法である．

3. δ-LFIT

ここでは本研究での提案手法 δ-LFIT について説明する．δ-LFIT は簡単に言うとルール分類器として見なすことが可能．与えられた入力の解釈遷移の中に，あるルールが存在するか存在しないかを見分けることにより，結果的に NLP を学習することが可能となる．LFIT 問題を分類問題と見立て，δ-LFIT の損失関数が実際に解釈遷移内に存在するルールとモデルが予測したルールとの交差エントロピーとなる．その損失関数を勾配降下法を用いてモデルを学習させることが可能となる．

3.1 ルールの分類

ルールを分類するには，まずあるあらゆるルールを列挙しなければならない．エルブラン基底 $\mathcal{B}$ が与えられた時，式 3 の形をとるルールは制限なしでは無限にある．

まず，ルールを簡略化する演算を定義する．

定義 1 (ルールの簡略化) ルールは以下の演算で簡略化可能:

- $a \wedge a$ は a に簡略化する
- $\neg a \wedge \neg a$ は $\neg a$ に簡略化する
- $a \wedge \neg a$ および $\neg a \wedge a$ は $\perp$ に簡略化する

ここで a はアトムである．

$\perp$ はルールに空のボディと同等である $a \leftarrow$.

以上の定義を利用し，最小ルールを定義する．

定義 2 (最小ルール) 最小ルールとはこれ以上簡略化できないルールである．

これらの定義ではアトムの順番は無視できるものとしている．$a \wedge b$ も $b \wedge a$ もどちらも最小ルールである．

最小ルールのみ考慮することで，分類する対象のルールの数が有限となった．特に，同じアトムが 2 回以上出現することがないので，ルールの長さが最長 $\|\mathcal{B}\|$ となる．これにより，すべてのルールを列挙することが可能となった．

すべてのエルブラン基底 $\mathcal{B}$ について，すべてのルールを含む有限順序集合 $\tau(\mathcal{B})$ を定義する．分類問題では，各クラスがどのルールに対応しているかがとても重要である．従って，ルールから自然数への写像を定義する．

その定義の前，まずいくつかの予備定義を説明する．

定義 3 (ルール長) ルール $R \in \tau(\mathcal{B})$ の長さは $\|b(R)\|$ として定義する．

定義 4 (順序集合の要素のインデックス) 順序集合 S が与えられた時，要素 $e \in S$ のインデックスは $\sigma_S(e) = \|S_{<e}\|$ として定義され，$\sigma_S : S \mapsto \mathbb{N}$ そして $S_{<e} = \{x \mid x < e, x \in S\}$ である．

定義 5 (順序エルブラン基底) 順序エルブラン基底 $\mathcal{B}_o$ は $\mathcal{B}$ と同じ要素を含む，ただし順序エルブラン基底では $<$ の関係が成り立つ．

$\mathcal{B}_o$ の $<$ 関係は適当に決めてよい．しかし本論文では便宜上主に辞書順とします．ルールの集合 $\tau_l(\mathcal{B}_o) = \{R \mid \|b(R)\| \leq l, R \in \tau(\mathcal{B}_o)\} \subseteq \tau(\mathcal{B}_o)$ として定義し，ルール長 l 以下の全てのルールを含んでいる．そのルールの集合の大きさは以下の式によって得られる：

$$\|\tau_l(\mathcal{B}_o)\| = \begin{cases} 1 & l = 0, \\ \|\tau_{l-1}(\mathcal{B}_o)\| + \binom{n}{l} \times 2^l & l > 0. \end{cases}$$

ここで $n = \|\mathcal{B}_o\|$ はエルブラン基底の要素数であり $\binom{n}{k}$ は二項係数である．

また，ルール長がちょうど l の全てのルールを含む順序集合 $\tilde{\tau}_l(\mathcal{B}_o) = \{R \mid \|b(R)\| = l, R \in \tau(\mathcal{B}_o)\}$ を考えたとき，この集合内の順序関係はまず負アトムの位置によって決まる．負アトムを 1 で表し，正アトムを 0 として表した時，その順序は表された 2 進数の大小関係によって決まる．例として，$\{a, b, \neg c\}$ は $(0, 0, 1)$ になり，左が最下位ビットとして捉えた時に 100_2 として表すことができる．この関係性では，$\{a, b\} < \{a, c\} < \{\neg a, b\} < \{\neg b, c\} < \{\neg a, \neg c\}$ の順序が成り立つ．

次に，ルールのインデックス $\sigma_{\tau(\mathcal{B}_o)}(R)$ について説明する．ルール R のインデックスは以下の式によって得られる：

$$\sigma_{\tau(\mathcal{B}_o)}(R) = \|\tau_{l-1}(\mathcal{B}_o)\| + \sigma_{\tilde{\tau}_l(\mathcal{B}_o)}(R)$$

ここで $l = \|b(R)\|$ ルール R の長さである．

あるエルブラン基底における全てのルールのインデックスを表 1 に表す．

ひとつのルールにて，あるアトムは正か負として現れる，またはそのルールに現れないという 3 つの可能性を考えた時，n 変数のシステムは 3^n 個のルールがあるといえる．つまり δ-LFIT の分類クラスの数は $3^n \times n$ 個である．

	$l=1$	$l=2$	$l=3$
$\tau(\mathcal{B}_o)$	$0 \to \{\}$ $1 \to \{a\}$ $2 \to \{b\}$ $3 \to \{c\}$ $4 \to \{\neg a\}$ $5 \to \{\neg b\}$ $6 \to \{\neg c\}$	$7 \to \{a,b\}$ $8 \to \{a,c\}$ $9 \to \{b,c\}$ $10 \to \{\neg a,b\}$ $11 \to \{\neg a,c\}$ $12 \to \{\neg b,c\}$ $13 \to \{a,\neg b\}$ $14 \to \{a,\neg c\}$ $15 \to \{b,\neg c\}$ $16 \to \{\neg a,\neg b\}$ $17 \to \{\neg a,\neg c\}$ $18 \to \{\neg b,\neg c\}$	$19 \to \{a,b,c\}$ $20 \to \{\neg a,b,c\}$ $21 \to \{a,\neg b,c\}$ $22 \to \{\neg a,\neg b,c\}$ $23 \to \{a,b,\neg c\}$ $24 \to \{\neg a,b,\neg c\}$ $25 \to \{a,\neg b,\neg c\}$ $26 \to \{\neg a,\neg b,\neg c\}$

表 1: エルブラン基底 $\mathcal{B}_o = \{a,b,c\}$ のルールのインデックス

3.2 NLP 行列

δ-LFIT の出力は行列だが，ここではその行列を NLP に変換する手法について説明する．

エルブラン基底 $\mathcal{B} = \{a,b\}$ および下記の NLP が与えられた時，

$$a(t+1) \leftarrow a(t) \wedge b(t)$$
$$a(t+1) \leftarrow \neg b(t)$$
$$b(t+1) \leftarrow \neg a(t)$$

これは下記の行列として表すことができる：

$$\begin{array}{c} \\ a \\ b \end{array} \begin{array}{c} \begin{array}{cccccccc} \{a\} & \{b\} & \{\neg a\} & \{\neg b\} & \{a,b\} & \{\neg a,b\} & \{a,\neg b\} & \{\neg a,\neg b\} \end{array} \\ \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \end{array}$$

ヘッドに複数のルールが存在する時は論理和を取る意味であり，その時はヘッドの行で複数の 1 をマークすることとなる．

3.3 モデル

ここでは提案手法のモデルについて述べる．ニューラルネットワークのパターン認識を最大限に引き出すために，ニューラルネットワークの構造自体は変更なしで実装した．

主に，モデルに解釈遷移列を与え，その遷移列内に存在するルールを分類し識別したい．従って，本研究では LSTM を用いて解釈遷移列のパターン認識を行い，その後フィードフォワードニューラルネットワークを用いて分類した．そのアーキテクチャを図 1 で表す．

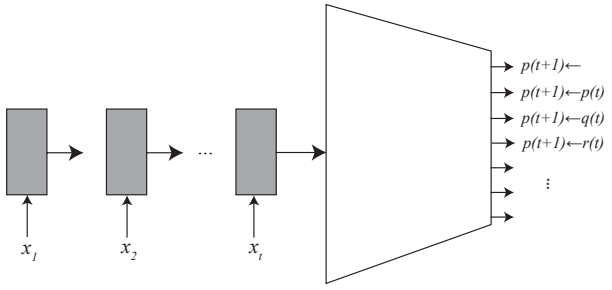

図 1: δ-LFIT アーキテクチャ

すべてのルール R に対して，ある遷移列 T が与えられた時，δ-LFIT は下記の確率を出力する：

$$p(R|T)$$

遷移列 T と一貫する論理プログラム P があるとし，δ-LFIT は P に存在するすべてのルール R について $p(R|T)=1$ を出力し，存在しないルールについて $p(R|T)=0$ を出力する．つまり，すべてのデータ Λ からサンプリングした時，(R,T) が実のラベル R と一致することを期待したい．従って，下記の負の対数尤度を最小化することが学習目的である：

$$\text{LFIT 損失} = -\mathbb{E}_{(R,T)\sim\Lambda}(R \times \log p(R|T) + (1-R) \times \log(1-p(R|T)))$$

上記した損失に加え，δ-LFIT にルール間の包含関係を学習させたいので，下記という包含損失を加える：

$$\text{包含損失}_S = -\sum_{R\in S}(R \times \log p(R|T) + (1-R) \times \log(1-p(R|T)))$$

簡単に言えば，互いに包含するようなルールはできる限り一つしか $p(R|T)=1$ とならないように設定している損失．

正則化などを加え，全体的な損失関数は下記の通りとなる：

$$\text{loss} = \text{LFIT 損失} + \lambda_1 \sum_{S\in\ \mathbb{S}} \text{包含損失}_S + \lambda_2 \|\theta\|^2$$

ここで λ_1 および λ_2 はハイパーパラメータであり各損失の重みを制御している．$\|\theta\|^2$ はすべてのパラメータに対する L_2-正則化である．

3.4 学習データの生成

良質な学習データを入手するのは，特に統計的機械学習においてかなり重要視されている [11]．δ-LFIT においては，分類をしているので，各クラスでの学習データが必要である．しかし各クラスはそれぞれルールに対応しているので，全てのルールにおいてそれなりの学習データが必要ということになる．現実的にそのような学習データを得るのは困難なため，本研究で用いる学習データは全て人工的に生成したものである．ここではその学習データの生成方法について説明する．

全てのルール $R \in \tau(\mathcal{B})$ について学習データを生成する．モデルの入出力に従い，学習データは (T,P) のペアであり，T は解釈遷移列そして P は R を含む最小の NLP である．学習データを生成するには，まず R を含む適当な P_o を生成する．この P_o からあらゆる初期値を与え解釈遷移列 T を作る．次にシンボリック手法の LFIT を用いて T から P_l を学習する．もし P_l の中に R が存在するなら，この (T,P_l) のペアは学習データとして加える．

4. 実験

まず提案手法が与えられた解釈遷移列からルールを分類できるか確認し，そして解釈遷移列にノイズを加え提案手法の有効性を確認する．本研究は Tensorflow [1] で実装し実験を行った．δ-LFIT を [10] から得られたそのうち 4 つのブーリアンネットワークにて有効性を確認した．4 つのブーリアンネットワークのうち，3 つが 3 変数のものであり 1 つが 5 変数である．

まず，各ネットワークについて，あらゆる初期値から生成される解釈遷移列をノイズなしに δ-LFIT に与え，δ-LFIT から予測で得られた NLP から，再びその元の解釈遷移列の復元を試みた．その復元した解釈遷移列と元の解釈遷移列との平均二乗誤差をとり評価した．

次に，元の解釈遷移列で 0 だったものを $[0, 0.5]$ の値へマッピングし，1 だったものを $[0.5, 1]$ の値へマッピングした．こ

のマッピングは正規分布に従いランダムに行った．0.5 が 0 および 1 両方からマッピングされることに注意したい，これによってモデルの曖昧さへの頑健性を調べる．また，本研究ではモデルをノイズに適応させるためにノイズの含む学習データで再学習させていない．

すべての実験結果を表 2 で表す．多くの場合ではノイズを含んだ入力では精度が落ちている．ただし唯一の例外が Raf のネットワーク．このネットワークでは安定状態が多く，ノイズなしではニューラルネットワークにとってパターン認識するのが困難であり，ノイズを加えたことで認識がうまくいったと考えられる．

ネットワーク	MSE (ノイズなし)	MSE (ノイズあり)
3-node (a)	0.095	0.137
3-node (b)	0.054	0.057
Raf	0.253	0.217
5-node	0.142	0.147

表 2: 予測した論理プログラムの平均二乗誤差

4.1 考察

本研究での主な目的はノイズを含むデータからその遷移列を説明できる NLP を学習することである．結果を見る限り，ノイズが含んだ時精度が落ちていたことからこの目標は果たしていない．しかし，学習データにノイズを入れ，学習時にもノイズへの対応を含ませることで精度が上がると考えられる．

そして，5 変数以上のネットワークには対応できなかった．主な原因として 5 変数以上のシステムの学習データを生成するには現実的な時間内では不可能だった．計算能力の向上を期待するとともに，マルチラベルマルチクラスの分類問題の分野におけるなにかの新展開も δ-LfIT の性能の向上につながると考えられる．

さらに，すべての実験において，解釈遷移列が安定状態である時，つまりすべての状態が同一の状態となっているとき，δ-LFIT は正しい NLP を予測することができなかった．特に Raf では安定状態が 5 つあり，性能の悪さが一目瞭然である．しかしノイズを加えることで，入力データに分散が出て，δ-LFIT にとって認識しやすくなったと考えられる．

5. 関連研究

関連研究として，Gentet らが提案した NN-LFIT 手法 [3] がある．この手法ではニューラルネットワークを学習したい対象のシステムをモデル化し，その後にルールをニューラルネットワークから抽出している．[3] では少量の学習データからでも精度の高い学習が可能であることを示したが，ノイズに対する頑健性は調べられていない．また，学習したいシステムがあるごとにニューラルネットワークを学習させる必要があるので，学習時間が無駄になるともいえる．

他に，Evans らによって提案された δ-ILP[2] がある．δ-ILP と本研究での提案手法の主な違いは，δ-ILP は微分可能な手法で ILP 問題を解くと同時に論理プログラムを出力することを目的としている．一方，本研究では LFIT 問題を微分可能手法で解かず，出力は論理プログラムのみである．

6. おわりに

本研究ではノイズの含んだ遷移列データから論理プログラムを学習するモデル δ-LFIT を提案した．LFIT をルールの分類問題に見立てたことより，学習は予測したルールと本当に存在するルールとの交差エントロピーを最小化することで実現した．特に，δ-LFIT では通常のニューラルネットワークモデルをただ用いて，変更なしに実装を行った．従って，今後ニューラルネットワーク技術について新たな進展があった時に，そのまま δ-LFIT の性能の向上につながると考えられる．そして，本研究でいくつか実験を行いその有効性を示した．ノイズを含んだ解釈遷移列についても δ-LFIT では論理プログラムの学習が可能であることを示した．今後の展開として，変数の数へのスケールアップできるアーキテクチャについて調べる．また，ノイズにさらに頑健とさせるため，学習データにあらかじめノイズを含ませて学習させその性能について調べる．

参考文献

[1] M. Abadi, A. Agarwal, P. Barham, et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] R. Evans and E. Grefenstette. Learning explanatory rules from noisy data. *CoRR*, abs/1711.04574, 2017.

[3] E. Gentet, S. Tourret, and K. Inoue. Learning from interpretation transition using feed-forward neural network. In *Proceedings of ILP 2016, CEUR Proc. 1865*, pages 27–33, 2016.

[4] K. Inoue, T. Ribeiro, and C. Sakama. Learning from interpretation transition. *Machine Learning*, 94(1):51–79, 2014.

[5] D. Martínez, G. Alenyà, T. Ribeiro, K. Inoue, and C. Torras. Relational reinforcement learning for planning with exogenous effects. *Journal of Machine Learning Research*, 18(78):1–44, 2017.

[6] T. Ribeiro and K. Inoue. Learning prime implicant conditions from interpretation transition. In *ILP 2015*, pages 108–125. Springer, 2015.

[7] T. Ribeiro, M. Magnin, K. Inoue, and C. Sakama. Learning multi-valued biological models with delayed influence from time-series observations. In *2015 IEEE 14th International Conference on Machine Learning and Applications (ICMLA)*, pages 25–31, Dec 2015.

[8] D. Rolnick, A. Veit, S. Belongie, and N. Shavit. Deep learning is robust to massive label noise, 2018.

[9] M. L. Seltzer, D. Yu, and Y. Wang. An investigation of deep neural networks for noise robust speech recognition. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 7398–7402. IEEE, 2013.

[10] A. Streck, H. Siebert, and H. Klarner. PyBoolNet: a python package for the generation, analysis and visualization of boolean networks. *Bioinformatics*, 33(5):770–772, 12 2016.

[11] C. Sun, A. Shrivastava, S. Singh, and A. Gupta. Revisiting unreasonable effectiveness of data in deep learning era. *CoRR*, abs/1707.02968, 2017.