.Deb

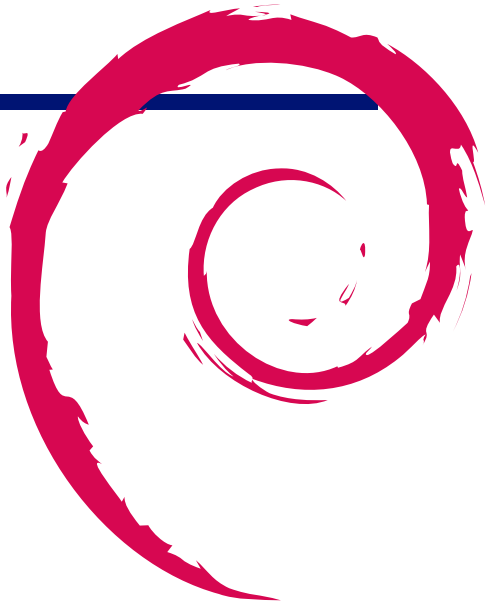
銀河系唯一のDebian専門誌

2020 年 7 月 18 日

ニューラルネットワーク特集

デビアン勉強会

目次



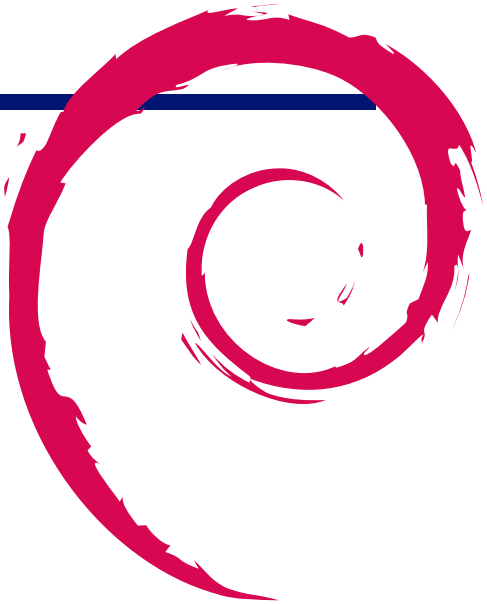
1　最近の Debian 関連のミーティング報告

杉本　典充

1.1　2020 年 6 月度 東京エリア・関西合同 Debian 勉強会

2020 年 6 月 29 日 (土) に東京エリア Debian 勉強会と関西 Debian 勉強会の合同でオンラインによる Debian 勉強会を開催しました。参加者は 12 名でした。

セミナーは「2020 MiniDebConf Online 共有会」として、参加者が初めて開催された MiniDebConf のセミナービデオを見て学んだこと、感じたことを共有しました。



2 事前課題

杉本 典充

今回の事前課題は以下です。

1. ニューラルネットワークについて聞いてみたいことがあれば教えてください

2.1 dictoss

1. (回答なし)

2.2 yy_y_ja_jp

1. (回答なし)

2.3 kare-zeri-123

1. (回答なし)

2.4 NOKUBI Takatsugu (knok)

1. (回答なし)

2.5 kenhys

1. (回答なし)

2.6 Kazuhiro NISHIYAMA (znz)

1. (回答なし)

2.7 yosuke_san

1. (回答なし)

2.8 ysaito

1. (回答なし)

2.9 ipv6waterstar

1. (回答なし)

2.10 jsynth21

1. (回答なし)

2.11 KatsunoriNoma

1. (回答なし)

2.12 gdevmjc

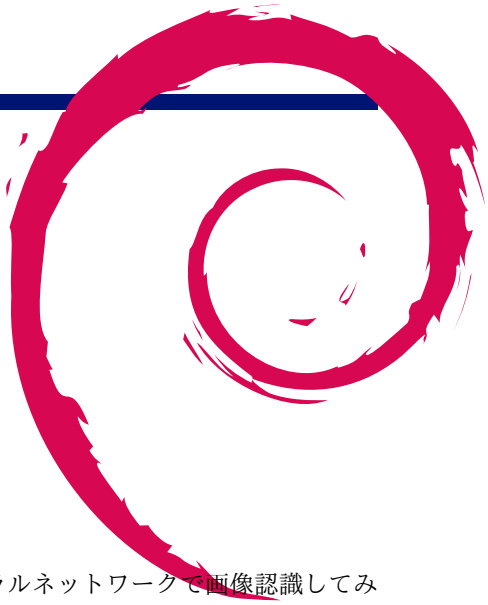
1. この 20 年に学問的に進化した点、現在学習の計算するためのベストプラクティスが聞ければ幸いです

2.13 koedoyoshida

1. 直接関係ないですが、「行列のできない過去の自分への覚書」が非常に面白かった `https://ishibaki.github.io/blog/ForLinearAlgebraBeginners` ほかに近い分野で知的に面白い物が有れば

2.14 dancerj

1. (回答なし)



3　10 年ぶりのニューラルネットワーク

野首貴嗣

過去、2010 年 3 月に行われた第 62 回東京エリア Debian 勉強会では「ニューラルネットワークで画像認識してみた」というタイトルで本庄弘典さんが発表をされていました。

それから 10 年が経過した 2020 年 7 月現在、ニューラルネットワークをとりまく状況がどのように変化したかを俯瞰的に見ていきます。

また、個人的に開発、公開している成果につても紹介します。

3.1　ニューラルネットワークのお気持ち

まず、ニューラルネットワークについての厳密ではない、なんとなくなお気持ちで概要を解説します。

パーセプトロン (https://ja.wikipedia.org/wiki/%E3%83%91%E3%83%BC%E3%82%BB%E3%83%97%E3%83%88%E3%83%AD%E3%83%B3) と呼ばれる、人間の脳 (ニューロン) を模した演算ユニットがあります。これは入力信号 X に対し、重み W を乗じバイアス b を加算した結果 Y をえる単純なユニットです。さらに、Y に対し活性化関数と呼ばれる関数を適用しその結果を分類タスクの結果として考えます。

分類が正しいなら 1 を、間違っているなら 0 を返す関数と考え、実際に用意したデータをもちいて訓練を行います。訓練は、逆誤差伝播 (バックプロパゲーション) という手法を使い、求められた結果と訓練データ上の正解との誤差を最小化するよう、重みを調整していきます。

単一のパーセプトロンは AND や OR を表現することができます。パーセプトロンの式は線形分離境界を表していると考えることで直感的に理解できるでしょう。さらには、XOR は実現できないことも理解できると思います (1 つの直線で分離できないため)。

ニューロンを増やしたり、多段に接続することで、より複雑な非線形分離が可能となります。この段数が概ね 4 段以上になったものをディープラーニングと呼ぶことが多いようです。

3.2　ブレイクスルー

2012 年に、Google が”Using large-scale brain simulations for machine learning and A.I.”(https://blog.google/topics/machine-learning/using-large-scale-brain-simulations-for/) という blog 記事と論文 (http://static.googleusercontent.com/media/research.google.com/ja//archive/unsupervised_icml2012.pdf) を公開しました。16 コアをもつマシン 169 台を駆使し、大量の画像を訓練することで教師なしで人体や猫の特徴を獲得したと話題になりました。このときに採用されたアーキテクチャは AutoAEncoder と呼ばれるものです。

AutoEncoder は入力側と出力側が等しくなるように構成された多段ニューラルネットワークで、入出力画像よりも小さなベクトル表現によって、入力画像の特徴を獲得することができるものです。AutoEncoder が行っていることは、一種の次元圧縮ともいえます。ニューラルネットワークを用いない次元圧縮手法もありますが、多段ニューラル

ネットワーク (ディープラーニング) が実用になることを示した大きな一歩でした。

3.3 ブレイクスルー実現要素

従来のニューラルネットワークが抱えていた課題を解決した要素にはさまざまなものがあります。

計算資源の向上

私がニューラルネットワークに触れた 1995 年の時点で、実際に利用されていたコンピュータのスペックは次のようなものでした。

- PA-RISC 25Mhz
- メモリ 16MB

そもそも利用可能なメモリが少なく、大きなニューラルネットワーク構造を持つこと自体が困難でした。現代においてはスマートフォンでも 4GB 程度のメモリを搭載するものも珍しくありません。

GPGPU の一般化

CPU 自体も当時と比較してかなり高速にはなりましたが、単一コアあたりの処理能力向上には限界が見えつつあります。一方でグラフィックス処理向けに設計された GPU をより一般的なコンピューティングに応用する環境が整備されてきました。

GPU は特に並列計算を得意としており、多段ニューラルネットワークの演算とは非常に相性がよく、その方向へ進化していっています。

大規模データセットの拡充

Google の猫の例で活躍したのは、Google photos や Facebook album 等、画像データを大量に収集できるプラットフォームの存在でした。

さらに、ラベル (物体の種別情報) を付与したデータセットも増えていきます。これにはクラウドソーシングサービスの拡充 (Amazon Mechanical Turk, Cloudworks 等) も寄与しています。

技術的な進歩

多段ニューラルネットワークが抱えていた他の問題として、勾配消失/発散問題がありました。逆誤差伝播をする際に、層が増えることによって伝播すべき誤差がどんどん小さくなってしまったり、逆に大きくなったりする問題です。

要因の一つに、活性化関数があります。シグモイド関数を使うことが昔は一般的でしたが、この関数が取りうる値は 0〜1 しかありません。より幅広い値を扱える活性化関数 (ReLU 等のランプ関数) の出現により、より大きな勾配を扱えるようになりました。

また勾配のクリッピングによって、勾配爆発を抑えることができるようになりました。その他 Batch Normalization などの正規化手法も学習の安定化に寄与しています。

分散表現の活用

単語や抽象的なカテゴリーといったものを表現するために分散表現 (埋め込み表現) が活用できることがわかり、自然言語処理などへの活用が広がりました。word2vec は単語埋め込み表現の最たる活用例です。

データ拡張

既存のデータを変形したり、ノイズをのせるなどのデータ拡張技術が主に画像認識分野で発達しています。これにより、学習データを水増しすることができ、認識制度向上に寄与しています。

教師データを必要としない学習

一般的な教師あり学習には正解データを用意する必要があり、その作成にはコストがかかります。一方で word2vec のような、教師データを必要としない手法が発達してきています。

既存のデータの一部をマスクしたり、並べ替えたりして正解を予想させることで、対象となるデータの特徴をうまく表現する情報を獲得することができます (表現学習)。

敵対的生成ネットワーク

生成モデルと識別モデルを組み合わせ、互いを競わせるように学習させることで、現実には存在しない画像な

どを生成する手法が提案され、広く使われています。

3.4 環境

データサイエンティストブーム

2000 年代よりデータサイエンティストブームが発生し、ゼロ年代は統計的手法が広まりました。その流れは 2010 年台でも継続し、深層学習手法にも波及しています。数多くの MOOCs プラットフォームも立ち上がり、安価で良質な学習環境が整備されています。

コンペティションの隆盛

Kaggle, SIGNATE といったコンペティションサイトが立ち上がり、技術者の研鑽が日々なされています。

実行環境の充実

かねてから GPGPU 環境は AWS, GCE などで提供されていましたが、それ以外にも無料で GPU の利用が可能な Google Colaboratory といったサービスが使えるようになり、ブラウザさえあれば環境構築不要で機械学習が行えるようになりました。

3.5 もたらされた変化

人を越える精度/能力

画像認識分野では、ImageNet(`http://image-net.org/`) というデータセットでいかに高い認識精度を達成するか、という試みが行われています。2012 年にディープラーニングベースの手法 AlexNet が登場して以来、急速にその精度は向上し、2015 年には ResNet によって人間の認識能力を超える精度を達成しました (参考: `https://www.slideshare.net/ren4yu/ss-234439652/21`)。

自然言語の分野でも、Stanford Question Answering Dataset (SQuAD) (`https://rajpurkar.github.io/SQuAD-explorer/`) などで人間の正答率を超える正答率を達成しています。

問題を分類タスクに落とし込むことができ、十分な学習データを用意することができれば、機械学習モデルは人間を超える精度を達成できる可能性を示しています。

生成モデル

たくさんの画像やテキストデータ、音楽データから、新しいデータを生成するモデルが広く使われるようになりました。

主に敵対的生成ネットワークによる成果ですが、テキストに関しては言語モデルベースの手法 (GPT-2 等) が主流です。

さらにこれらを応用することで、画像復元 (inpainting) や自動着色、さらには描画支援 (GauGAN) といったことが実現できています。

機械翻訳

ディープラーニング以前の機械翻訳は統計ベースの手法が利用されていました。この手法はあまり流暢ではない文が生成されがちという欠点があったのですが、ニューラル機械翻訳によって、その問題は大きく改善されました。

一方で GPU が持つメモリがボトルネックとなり、語彙を多くもたせることが難しくなりました。そのため、単語より小さな単位 (サブワード) でトークンを区切ることによって語彙を抑える手法が主流となってきています。

ニューラルネットワークの学習は多くが end-to-end で行えるため、正確な単語区切りや品詞を考慮しなくても良くなったという事情もあります。

その副作用として、多言語を一度に扱えるようにもなりました。サブワードより小さい、バイト単位で区切る手法もあります。この場合、デコード時に正しいバイト列であることを保証する必要性があります。

3.6 新たな課題

ニューラルネットワーク固有の課題も登場しています。

翻訳ミス

統計ベースの機械翻訳でも翻訳ミスはありましたが、ニューラル機械翻訳の場合、それを見つけることが難しくなっています。なまじ流暢な結果を出力してしまうため、注意深く見ないと訳抜けなどを見逃してしまいがちです。また、統計ベースでは、原文と翻訳文のフレーズ対応が処理の関係上あきらかだったのですが、ニューラル機械翻訳では end-to-end な学習になったことで、翻訳文と原文との対応づけが明らかでなくなったというデメリットもあります。

他に、逆の意味を出力してしまうことがある問題もあります。これも訳文の対応付けが不明瞭なため、原因をさぐることは難しい問題です。

同じ文言を繰り返し生成するという、ニューラル機械翻訳特有の問題もあります (参考事例: `https://www.slideshare.net/ToshiakiNakazawa/nlp2017-nmt-tutorial/79`)。

データセットとバイアス

近年よく問題としてとりあげられるものに、バイアスがあります。これは、学習に使ったデータセット内に偏りがあった場合、特に注意をしないとそれがそのまま機械学習モデルに反映されてしまいます。

たとえば、多くの場合英語で”sister” という単語は姉・妹の区別をつけることはあまりなく、機械翻訳でそれを日本語にした際に、文脈に応じて出現しやすい方に訳してしまうことが発生します。

敵対的サンプル

人間が見るとパンダなのに、識別モデルにはテナガザルに見えるような画像が有名です (参考: `https://speakerdeck.com/settenqb/ji-jie-xue-xi-tosekiyuritei?slide=5`)。このように機械翻訳モデルを騙す画像を生成する技術と、それを防衛する技術が互いにしのぎを削っています。

バックドア

訓練に用いるデータセットに悪意あるデータを混ぜることで、バックドアを仕掛けるという手法もあります。クラウドソーシングなどを使う際には、用途によってはそのような問題があることを想定しておく必要もあるでしょう。

3.7 フレームワーク

よく使わていた (いる) 代表的なフレームワークには以下のものがあります。

- 古参
  - Torch7
  - Caffe
  - Theano
- 現代的なもの
  - Chainer
  - PyTorch
  - TensorFlow
  - Keras
  - Darknet

どれも多次元配列を扱うことができ、自動微分機能を備えています。

3.8 Web/JavaScript ランタイム

先に述べたようなフレームワークで訓練されたモデルを、ブラウザや JavaScript で実行するフレームワークがいくつかあります。

- WebDNN
  - https://mil-tokyo.github.io/webdnn/ja/
  - 日本発
  - 複数のフレームワークに対応
  - 現在開発は inactive
- TensorFlow.js
  - https://www.tensorflow.org/js
  - TensorFlow 公式
  - 訓練もサポート
  - 多くの訓練済みモデルを配布
  - https://github.com/tensorflow/tfjs-models
- ONNX.js
  - https://github.com/microsoft/onnxjs
  - フレームワーク共通フォーマット ONNX に対応
  - 対応している演算に制限あり

3.9 作成したものの紹介

WebDNN を用いて、いくつかのデモサイトを公開しています。

edges2cats
: Chainer と pix2pix を用いて、猫の線画に着色をするモデルを公開しています (`http://pix2pix.daio.net/`)。公式のデモサイト `https://affinelayer.com/pixsrv/` でも同じことができますが、こちらは猫着色モデルのみ、訓練データを公開していません。それ以外のデモは訓練データを含め公開しているので再現可能です。
同じことをするために、猫画像収集スクリプトを公開しています `https://github.com/knok/pixabay-cat-images`。
猫画像から背景の除去は DilatedNet の keras 実装 `https://github.com/nicolov/segmentation_keras` とその訓練済みモデルを用いています (解説記事: `https://qiita.com/knok/items/6ad09cc870739dbd921b`)。

Wasserstein AutoEncoder
: TensorFlow で実装されていた著者実装 `https://github.com/tolstikhin/wae` をもとに、Chainer へ移植し、2 次元の潜在空間へ 300 程度のイラストを埋め込み、推論を行えるサイトとして公開しました `http://wae-friends.daio.net` 。

また、TensorFlow.js と訓練済みモデルを利用したデモサイトも公開しています。

バーチャル背景
: 訓練済み BodyPix モデル `https://github.com/tensorflow/tfjs-models/tree/master/body-pix` を用いて Web カムから人体部分のみを抽出 (セマンティックセグメンテーション) し、背景用画像と合成して表示するサイトを GitHub Pages で公開しています `https://knok.github.io/virtbg/` (ソースコード:

https://github.com/knok/virtbg)。
これ単体では配信などには使えませんが、OBS Studio のウィンドウキャプチャー機能と OBS-v4l2sink を組み合わせれば、v4l2loopback を利用可能な任意のサービスに応用できます (解説記事: https://qiita.com/knok/items/b3eb87769151ac04efeb)。

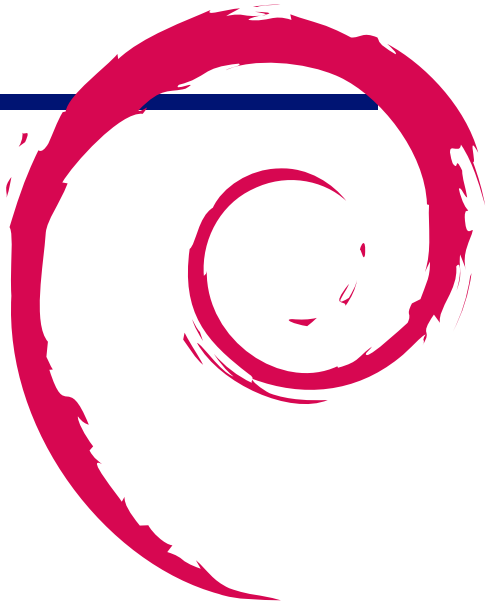
VRM Thee.js PoseNet Sample
既存の公開されていたコード https://github.com/t-takasaka/vrm-three-posenet を fork し、最近の API に合わせつつ再配布可能な VRM モデルに差し替えたものを公開しています https://knok.github.io/vrm-three-posenet/ (ソースコード: https://github.com/knok/vrm-three-posenet)。
こちらは訓練済み PoseNet https://github.com/tensorflow/tfjs-models/tree/master/posenet を利用しています。

3.10 Debian における訓練済みモデルの扱い

2020 年 7 月時点では、ニューラルネットワークの訓練済みモデルについて、まだ明確なポリシーは決まっていません。ドラフトレベルのものはあります https://salsa.debian.org/deeplearning-team/ml-policy。再現性のため、訓練データや訓練用コードを含んだ状態が望ましいという方向性で進みつつあります。
その観点でいうと、TensorFlow Hub や tfjs-models はその条件を満たしていません。そういったものを用いるツールはおそらく main として配布されないのではないかと個人的には思っています。
一方で、Deep でない機械学習訓練済みモデルは既に配布されている現状があります (例: OpenCV の識別モデル https://salsa.debian.org/science-team/opencv/-/tree/master/data)。機械学習という分野自体でも再現性は重視されてはいますが、公開されている多くのものは研究用途に限定したデータセットや訓練済みモデルなのが現状です。プライバシーとの兼ね合いもあり、この問題の解決は難しいと思われます。

3.11 まとめ

2010 年の勉強会から 10 年が経過し、ニューラルネットワークをベースとした機械学習手法は飛躍的な発展を遂げました。今では Web ブラウザやスマートフォンなどのモバイルデバイスで動作させることも容易になってきています。
一方でニューラルネットワーク固有の問題や、データセットに起因する諸問題 (バイアス、プライバシー等) もあります。
DFSG に照らし合わせると、再現性を担保するようなデータセットや訓練コードを伴った訓練済みモデルの配布が望ましいところですが、その実現はなかなかむずかしそうです。



4　メモ

Debian 勉強会資料

2020 年 7 月 18 日　　初版第 1 刷発行

東京エリア Debian 勉強会 （編集・印刷・発行）