Deep Learning 輪読会 2017

第12章　アプリケーション

2017.12.11

東京大学工学部システム創成学科 松尾研究室

B4 久保静真

構成

12.1 大規模深層学習

12.2 コンピュータビジョン

12.3 音声認識

12.4 自然言語処理

12.5 その他のアプリケーション

12.1 大規模深層学習

- ニューラルネットワークの性能向上
  - 1980年代と比較して精度が飛躍的に向上し、より複雑なタスクに適用可能に。
  - 重要な要素のひとつに使用するネットワークの劇的増加がある。

    土台となるハードウェア・ソフトウェアに高い性能が求められる。

12.1.1 高速なCPU上での実装

- CPU上での効率的な実装
  - 2011年最高性能のCPUでは浮動小数点演算より固定小数点演算のほうが高速。
  - ただし、CPUのモデルによっては特性が異なり上記と逆になることもある。

    結局CPU個々に特化した調整が必要、、

- 機械学習に従事する研究者はこれらの実装に無頓着であるが、実装性能がモデルのサイズを制限し、モデル精度の低下につながる。

12.1.2 GPU上での実装

- GPU (グラフィカルプロセッシングユニット)
  - 元々はグラフィックアプリケーション用 (特にビデオゲーム)
  - システムのメモリ帯域幅がCPUに対して圧倒的利点となる。
  - ニューラルネットワークは複雑な制御が少なく独立して処理できる多数なニューロンに分けることが出来るためGPUの並列計算の恩恵を受けやすい。
- GPGPU(汎用GPU)
  - 描画用のサブルーチンだけではなく、任意のコードを実行出来るようになる。
  - GPU利用で考慮すべき点は、コアレスド（近いスレッドが近いアドレスを同時にアクセスするが効率的）とワープ（同じ命令を実行するグループ）である。
  - 高い性能を持つGPU用のコードを書くのは困難なため、実際には高性能演算用のライブラリを構築・利用する。(Theano、TensorFlow、Torchなど)

12.1.3 大規模分散処理を利用した実装

- データ並列処理
  - 入力事例ごとに分散処理（推論は簡単だが、訓練は多少困難）
  - 訓練時は標準の勾配降下の定義は逐次的なアルゴリズムであるため非同期勾配降下法を使う。
  - 非同期勾配降下法では複数のプロセッサがメモリ上のパラメータを共有する。
- モデル並列処理
  - 複数の計算機がそれぞれモデルの異なる部分を実行

12.1.4 モデル圧縮

- 商用アプリケーションへの適用
  - 機械学習を利用する多くの商用アプリケーションでは推論の際の実行時間と使用メモリが少ないことが訓練のときよりもはるかに重要になる。
  - パーソナライゼーションを利用しないモデルでは一度訓練してしまえば、それを配布して多くのユーザーが使うことが出来る。

    推論のためモデル圧縮したい！
- モデル圧縮
  - 推論に要するコストを削減するための重要な戦略
  - 元のモデルのサイズが主に過剰適合を防止する目的で決定された場合に適用出来る。
  - 参考(モデル圧縮): https://nico-opendata.jp/ja/casestudy/model_compression/index.html
  - 書籍の中では蒸留 (Hinton et al., 2015) に触れられているよう。

12.1.5 動的構造

- 動的構造(dynamic structure)
  - データ処理の高速化のための戦略。並列性が低下するのが欠点。
  - 多くのニューラルネットワークのうちどの部分集合を実行すべきかを入力データに基いて動的に決定出来る。個々のニューラルネットワークもどの特徴量の部分集合を使うか動的に決定できる。（条件付き計算）
- カスケード(cascade)
  - 分類器の推論処理の高速化の戦略。稀に出現する対象の検知に適用出来る。
  - 高い再現率を持つモデルと高い適合率を持つモデルの2段構えにする。(最初のモデルで対象を含まない入力を除外)
- ゲーター(gater)
  - いくつかのエキスパートネットワークのうちどれを使うかを選択。ゲーターが出力するのは各要素の出力の重みで、重みをつけて混合する混合エキスパートと1つのエキスパートを選択するハード混合エキスパートがある。

12.1.6 専用ハードウェアによる深層ネットワークの実装

- ニューラルネットワーク専用ハードウェア
  - 過去数十年間、ASIC(特定用途向け集積回路)として開発されてきた。
  - 最近はより柔軟なFPGAの実装が開発されてきている。

- ニューラルネットワークに必要な浮動小数点の精度
  - bit数を減らすことが出来ればハードウェア表面積、電力要件、乗算の実行に必要な計算時間が削減出来る。
  - 汎用処理装置上では通常32bit,64bitの精度が使われるが、もっと精度が低くても問題なく利用できることは以前から知られていた。（特に推論時）
  - 誤差逆伝播を利用する深層ニューラルネットワークの利用や訓練には8bitから16bitの精度で十分なことが示唆されている。

12.2 コンピュータビジョン

- 深層学習の最も一般的な標準ベンチマークタスクは、物体認識または光学的な文字認識の形式。
- コンピュータビジョンの深層学習では人間の能力を複製することを目的とした狭い範囲のAIの中核技術に集中している。
- 生成モデルが深層学習研究の指針の1つになり画像合成に関する研究も数多く行われているが、無から画像を合成しても通常コンピュータビジョンの取り組みとはみなされない。
- 画像の欠損を修復したり、画像から物体を取り去ったりするコンピュータビジョンのタスクの画像復元には画像合成可能なモデルは一般に有用である。

12.2.1 前処理

- 画像に必要な前処理
  - コンピュータビジョンの分野では前処理の必要性は比較的小さい。
  - 厳格に守るべき前処理は同じスケールを持つように画像を整形することだけ。

- 前処理の種類

1. 各事例を標準的な形式に処理
   - モデルが考慮すべき変動量を抑えるため。
   - 大規模なデータがあるときや大きなモデルを使うときは前処理の多くは不要な場合が多い。例えば、AlexNetのシステムでは前処理として訓練事例の各ピクセルから平均値を引くという処理しかしていない。
2. データ集合拡張:
   - 少しずらしてクロッピングなど。汎化誤差低減のため。

12.2.1.1 コントラスト正規化

- 画像ごとにコントラストがばらつかないようにするため、各画像から平均値を差し引いた後画素の標準偏差が定数sに等しくなるように大きさを変更する。

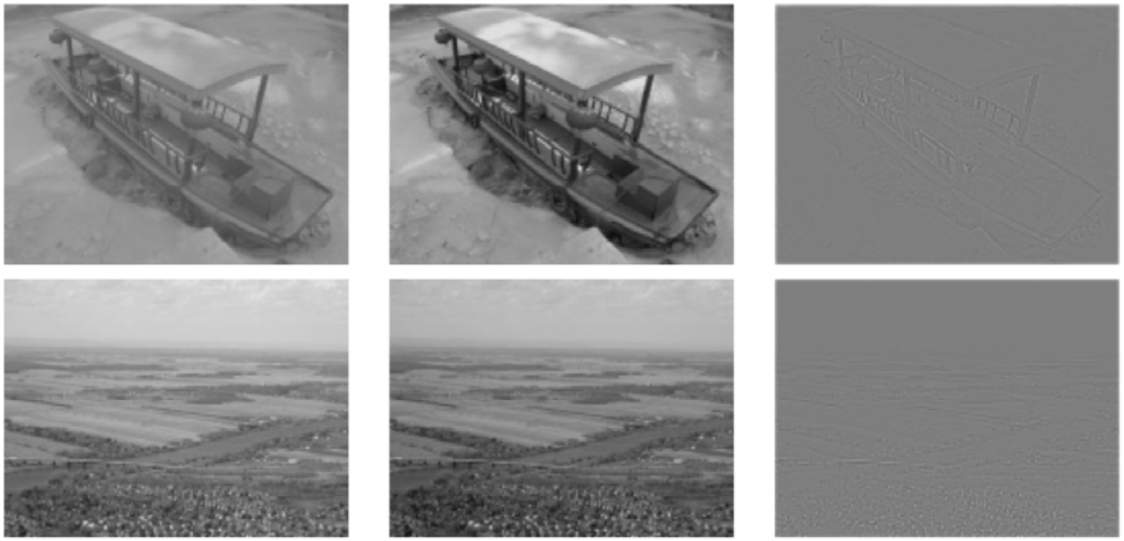
- 大域コントラスト正規化 (GCN)

$$X'_{i,j,k} = s \frac{X_{i,j,k} - \bar{X}}{\max\left\{\epsilon, \sqrt{\lambda + \frac{1}{3rc}\sum_{i=1}^{r}\sum_{j=1}^{c}\sum_{k=1}^{3}\left(X_{i,j,k} - \bar{X}\right)^2}\right\}}.$$

- 局所コントラスト正規化 (LCN)
  - 大域コントラスト正規化ではエッジやコーナーなど際立たせたい画像の特徴を強調出来ないことがよくある。（大きな暗い領域と大きな明るい領域があるとき）
  - 大域コントラスト正規化とは違い小さなウィンドウごとに正規化する。
  - 各ピクセルの値から周辺ピクセルの平均値を引き、周辺ピクセルの標準偏差で割る。

12.2.1.1 コントラスト正規化

- GCNとLCNの視覚的な比較



- 局所コントラスト正規化は微分可能な演算で前処理だけではなく、ネットワークの隠れ層に適用される非線形変換としても利用出来る。

12.2.1.2 データ集合拡張

- 訓練事例のコピー
  - ランダムな移動や回転、反転など。
  - 画像の色のランダムな摂動（小さな補正）
  - 非線形幾何学的歪み

12.3 音声認識

- 自動音声認識

$$f^*_{\mathrm{ASR}}(\boldsymbol{X}) = \arg\max_{\boldsymbol{y}} P^*(\mathbf{y} \mid \mathbf{X} = \boldsymbol{X}) \qquad \begin{aligned} \boldsymbol{X} &= (\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(T)}) \\ \boldsymbol{y} &= (y_1, y_2, \ldots, y_N) \end{aligned}$$

P*は入力Xを目標yに近づける真の条件付き分布

- (ニューラルネットではない)音声認識
  - 隠れマルコフモデル(HMM)とガウス混合モデル(GMM)を統合したもの。
  - GMM: 音響特徴と音素の特徴との関連のモデリング
  - HMM: 音素の系列をモデリング
  - GMM-HMMモデル: HMMによって音響波形を音素と離散副音素状態（各音素の開始、中間、終了など）の系列生成をし、GMMが各離散記号を音声波形の短いセグメントに変換。
  - TIMIT: MNISTのような音声認識のベンチマーク

12.3 音声認識

- 音声認識の歴史
  - 1980年代~2009年、2012年では、GMM-HMMシステムが支配的。ただし1980年代末期から1990年代初頭にかけてニューラルネットを利用した性能もほぼ同等かむしろ上回る。(TIMIT 誤り率26%) (ニューラルネットワークに切り替えるに値する説得力のある理由が見いだせなかった)
  - 2000年代末期まではGMM-HMMシステムに追加する特徴量をネットワークで学習。
  - 2009年からは制限付きボルツマン(Ⅲ章)のモデルを利用して20.7%の誤り率へ。
  - 最終的にはボルツマンマシンに基づくものからReLUとドロップアウトのような技術に移行。畳み込みネットワークの利用もある。
  - 現在進行中の流れとして、HMMを取り除いたend-to-endの深層学習のシステムである。LSTM RNNを学習するのにMAP推定を利用。誤り率17.7%へ。

12.4 自然言語処理

- コンピュータが英語やフランス語などの人間の言語を使用するための技術。

12.4.1 n-gram

- 最も初期に成功したモデルはn-gramと呼ばれる固定長のトークンの系列のモデルにもとづいていた。1つのn-gramはn個のトークンの系列。
- n番目のトークン付き条件確率をその前のn-1個で表す。より長い系列の確率分布は以下のように定義する。

$$P(x_1, \ldots, x_\tau) = P(x_1, \ldots, x_{n-1}) \prod_{t=n}^{\tau} P(x_t \mid x_{t-n+1}, \ldots, x_{t-1}).$$

- 通常はn-gramとn-1gramを同時に学習する。

$$P(x_t \mid x_{t-n+1}, \ldots, x_{t-1}) = \frac{P_n(x_{t-n+1}, \ldots, x_t)}{P_{n-1}(x_{t-n+1}, \ldots, x_{t-1})}$$

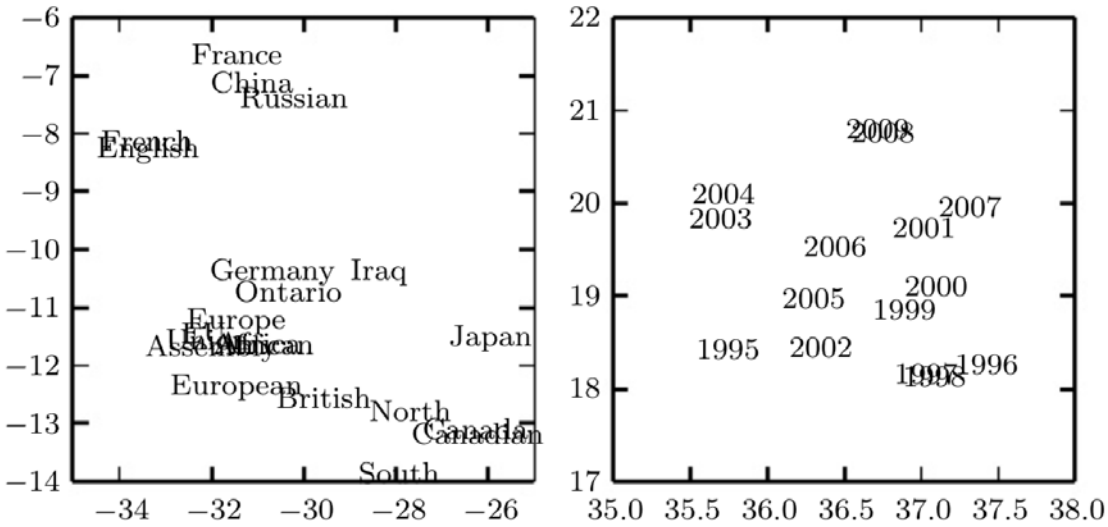
- 特にn=1のときはユニグラム、n=2のときはバイグラム、n=3のときはトライグラム。

12.4.1 n-gram

- 平滑化
  - 訓練集合の回数によって推定されるPがゼロとなると致命的なのでそれを防ぐ。
- バックオフ法
  - 出現頻度が低すぎて高次のモデルが使えないときに低次のモデルを参照する。
- クラスベース言語アルゴリズム
  - n-gramだとどの2つの単語間も同じ距離。これの改善のために単語カテゴリの概念を導入。

12.4.2 ニューラル言語モデル

- 単語の分散表現を用いることで次元の呪いを克服するように設計された。（n-gramは次元の呪いに脆弱）
- 各単語を別の単語と区別して符号化する能力を失うことなく、2つの単語が似ていることを認識出来る。
- 単語埋め込み(word embedding)のこと。



12.4.3 高次元の出力

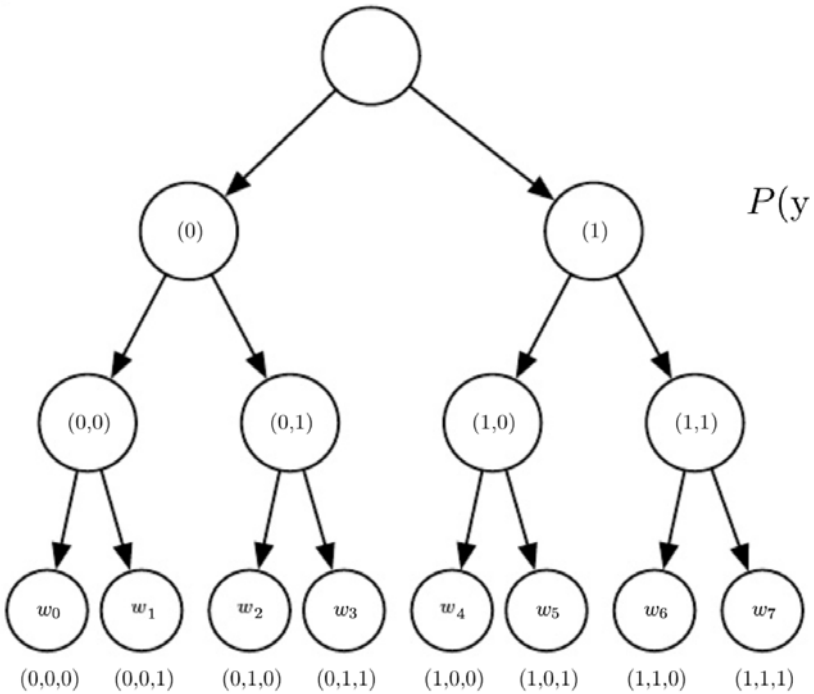
- 多くの自然言語アプリケーションでは出力の単位は単語が多く、数十万の単語集合が含まれる。
- 分布を表現するための単純なアプローチは、隠れ表現から出力空間へのアフィン変換を適用し、ついでソフトマックス関数を適用すること。
- しかし、単語集合が数十万になることから、高いメモリコストと計算コストが要求される。

12.4.3.1 ショートリストの利用

- 最初のニューラル言語モデル(Bengio et al, 2001, 2003)では語彙サイズを10,000語や20,000語に制限することにより対処していた。
- このアプローチを基に(Schwenk and Gauvain 2002, Schwenk 2007)では語彙集合を頻出語で構成されるショートリストとより低い頻度の単語で構成されるテールリストに分けた。ショートリストはニューラルネットでテールリストはn-gramで処理。
- 欠点はニューラルネットの汎化能力の利点が頻出単語に限定されるところ。

12.4.3.2 階層的ソフトマックス

- 計算的負担を軽減するために確率を階層的に分解する。
- V必要な計算量をlogVにまで減らすことが出来る。
- 実際には次のサンプリングベースの手法より結果が悪くなる傾向がある。



$$P(\mathrm{y} = w_4) = P(\mathrm{b}_0 = 1, \mathrm{b}_1 = 0, \mathrm{b}_2 = 0)$$
$$= P(\mathrm{b}_0 = 1)P(\mathrm{b}_1 = 0 \mid \mathrm{b}_0 = 1)P(\mathrm{b}_2 = 0 \mid \mathrm{b}_0 = 1, \mathrm{b}_1 = 0).$$

12.4.3.3 重点サンプリング

- 次の位置の現れることのないすべての単語からの勾配による寄与の計算を明確に避ける。
- 重点サンプリングでは、P(i | C)の計算が必要で全てのスコアaを計算する必要が残る。
- バイアス重点サンプリングで効率的な計算を実現。

$$
\begin{aligned}
\frac{\partial \log P(y \mid C)}{\partial \theta} &= \frac{\partial \log \operatorname{softmax}_y(\boldsymbol{a})}{\partial \theta} \\
&= \frac{\partial}{\partial \theta} \log \frac{e^{a_y}}{\sum_i e^{a_i}} \\
&= \frac{\partial}{\partial \theta} (a_y - \log \sum_i e^{a_i}) \\
&= \frac{\partial a_y}{\partial \theta} - \boxed{\sum_i P(y = i \mid C) \frac{\partial a_i}{\partial \theta}}
\end{aligned}
$$

$$
w_i = \frac{p_{n_i} / q_{n_i}}{\sum_{j=1}^{N} p_{n_j} / q_{n_j}}.
$$

$$
\sum_{i=1}^{|\mathbb{V}|} P(i \mid C) \frac{\partial a_i}{\partial \theta} \approx \frac{1}{m} \sum_{i=1}^{m} w_i \frac{\partial a_{n_i}}{\partial \theta}.
$$

12.4.3.4 雑音対照推定とランキング損失

- サンプリングに基づく他の提案手法もあり、そのひとつにランキング損失がある。

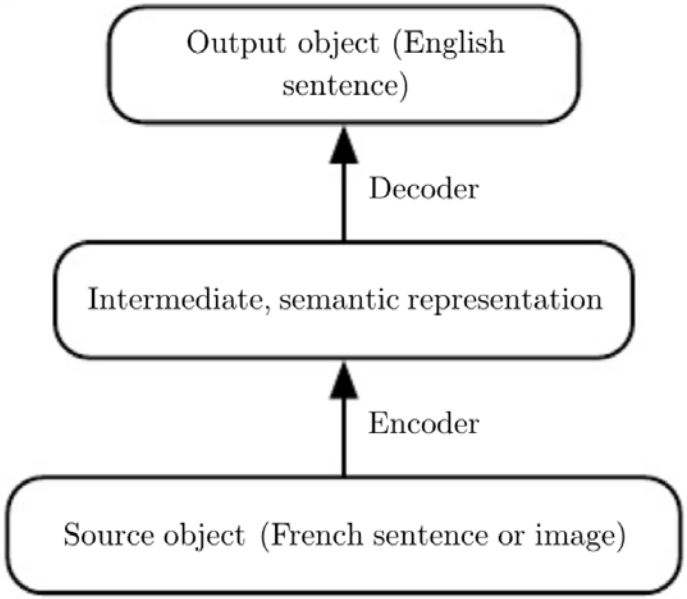
$$L = \sum_i \max(0, 1 - a_y + a_i).$$

12.4.4 ニューラル言語モデルとn-gramの統合

- ニューラルネットでn-gramを利用する利点
  - 1つの事例を処理するのにごく少量の計算しか要しない。一方で大きなモデルの容量を達成する。
  - ニューラルネットはパラメータ数2倍になると計算時間をおよそ2倍になる。
  - 容量を2倍にする簡単な方法の1つとして、ニューラル言語モデルとn-gram言語モデルからなるアンサンブル。

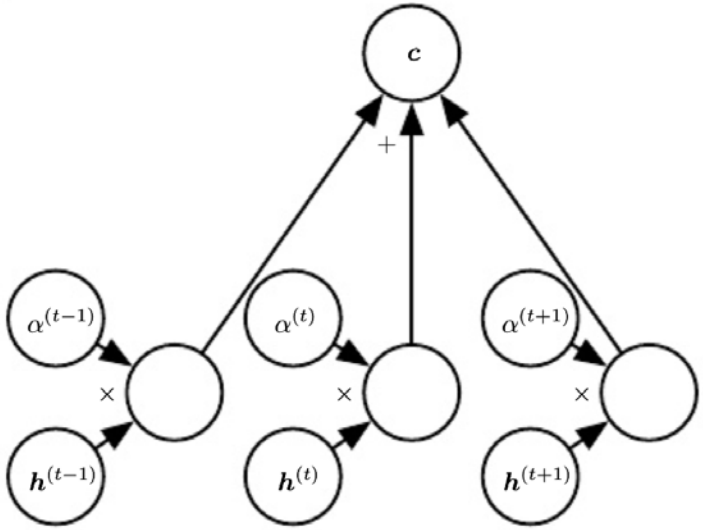
12.4.5 ニューラル機械翻訳

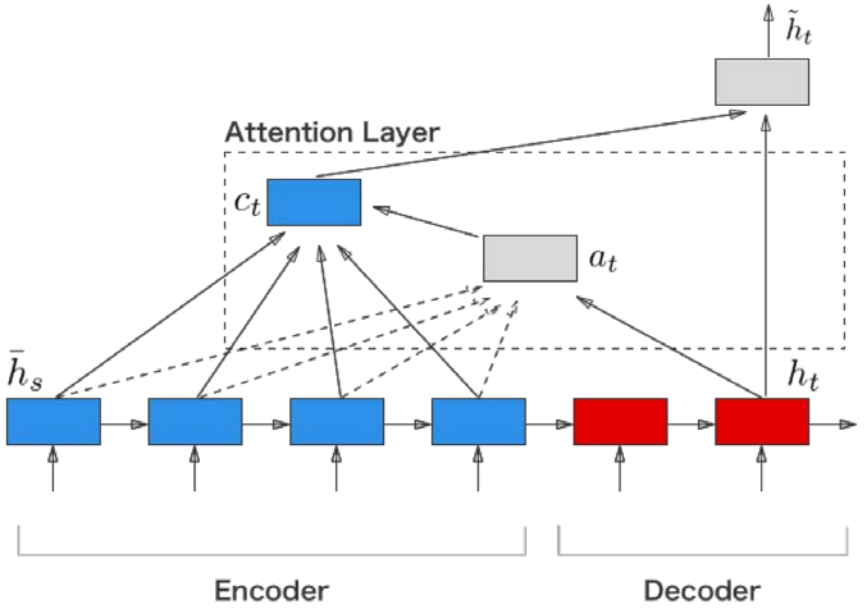
- MLPがn-gramによって算出される推定値にとってかわる。
- MLPを利用する欠点は系列を固定長にすることであったが、RNNがこれを解決した。
- 入力を読み取って、RNNや畳込みネットワークが文脈Cを生成し、それをもとにモデル（通常はRNN）がターゲット言語で文を生成する。



12.4.3.1 アテンションメカニズムの利用とデータ片のアライメント

- 非常に長い文章のすべての意味を詳細に捉えることは非常に困難。
- 効率的なアプローチとして、入力文の異なる部分に焦点を当てて、次の出力語を生成するための意味の詳細を収集する。
- 時間ステップごとに入力系列の特定の部分に焦点を当てるアテンションメカニズムと呼ばれる機構である。





12.4.6 歴史的視点

- シンボルの埋め込みという考え方から単語の埋め込みの考え方に拡張された。
- 基本単位を単語として言語モデリングの精度を向上させて来たが文字ベースの研究も進んでいる。
- t-SNE次元削減アルゴリズムの開発やそれを利用した可視化アプリケーションによって、埋め込みの2次元視覚化は言語モデルを分析するための一般的なツールとなった。

12.5.1 推薦システム

- 協調フィルタリング

- ユーザー1とユーザー2が似たような嗜好があるならユーザー1が好むアイテムはユーザー2も好むだろうと予測する。
- A: 各行にユーザーの埋め込み、B: 各列にアイテムの埋め込み、b,c:　各ユーザー、アイテムのバイアス

$$\hat{R}_{u,i} = b_u + c_i + \sum_j A_{u,j} B_{j,i}.$$

- 協調フィルタリングは新しいアイテムが入ったときに履歴がなく、評価方法がないのが問題
- コンテンツベース推薦システムにより、ユーザープロフィールや各アイテム情報も使用することが出来る。

12.5.1.1 探索と活用

- 文脈付きバンディット
- 探索と活用のトレードオフ

12.5.2.1 知識・関係・質問応答

- 2つのエンティティ間を捉えるために分散表現をどのように訓練出来るか。
- 物体に関する事実と物体同士がどのように相互作用するかを形式化

- アプリケーション
- リンク予測
- 語義曖昧性解消
- 質問応答システム

12 終わりに

- Ⅱ部
  - 深層ネットワークを用いる最新の実践とその中で使われている最も成功したすべての方法について記述。
  - 十分な訓練データがあれば強力なアプローチ。
- Ⅲ部
  - 研究の領域。課題は困難で今までのように解決には近づいていない。

参考文献 1 (GPU関連)

- GPUプログラミング・応用編
  - http://www.gsic.titech.ac.jp/supercon/main/attwiki/index.php?plugin=attach&pcmd=open&file=gpu-prog-2.pdf&refer=Supercomputing%20Contest%202012%2FGPU%E3%83%97%E3%83%AD%E3%82%B0%E3%83%A9%E3%83%9F%E3%83%B3%E3%82%B0%E8%B3%87%E6%96%99
- GPUプログラム構造の詳細(threadとwarp)
  - http://www.toffee.jp/streaming/gpgpu/advanced_gpgpu/2015/advanced_gpgpu03.pdf

参考文献 2

- Deep Learning
  - Ian Goodfellow, Yoshua Bengio, Aaron Courville
  - 日本語版
    https://www.amazon.co.jp/%E6%B7%B1%E5%B1%A4%E5%AD%A6%E7%BF%92-Ian-Goodfellow/dp/4048930621