Stacked Denoising Autoencoderを用いた語義判別

二輪 和博　馬　青
龍谷大学大学院理工学研究科数理情報学専攻

1　はじめに

自然言語処理における基礎的な課題の一つとして多義性解消の問題がある．例えば「頭」という言葉は体の部位を指すこともあれば，人の上に立つ者を指すこともある．すなわち，同じ表記でも文脈により語義が異なる場合がある．語義を間違った解釈をすれば，たとえば機械翻訳において正しい翻訳をすることは不可能となる．また，質問応答システムについても同様なことがいえる．このように，多義性解消は自然言語処理のさまざまな応用問題に強く関わっており必要不可欠な要素技術である．

本研究の目的は多義性解消に Deep Learning[1] を適用しその性能を調べることである．Deep Learning とは近年画像認識や音声認識など多くの分野で非常に高い精度を出し機械学習の分野にブレイクスルーを起こしているニューラルネットワークの一種である．実験には 2001 年に行われた SENSEVAL2 コンテスト日本語辞書タスク [2] のデータを使用した．他の学習器と性能比較するため Multilayer Perceptron (MLP) と Support Vector Machine (SVM) を使用した．

2　データ

2.1　データと素性

SENSEVAL2 コンテスト日本語辞書タスクは語義曖昧性解消に対するタスクであり，データは名詞 50 個動詞 50 個の計 100 単語，各単語について学習用，評価用のデータがそれぞれ 100 が事例含まれる．各事例にはそれぞれの単語が文中で使用された際の周辺の単語情報が含まれる．周辺単語の情報は文字列情報，RWC コーパスの品詞情報，分類語彙表の番号，JUMAN や KNP の解析結果による形態素情報や構文情報等の計 63 種の素性からなる．各事例に対しラベルデータ，すなわち正解となる語義 ID が割り振られたデータが与えられている．

2.2　ベクトル変換

Deep Learning 等の学習器を使用するためにそれぞれの事例を一つのベクトルに変換する．まず素性データのベクトル変換の方法について説明する．前処理として同じ単語内の全ての事例を調べて各素性の値について和集合をとる．それらの各事例について，素性値集合の各要素がその事例の素性値に含まれるなら 1，含まれないなら 0 として並べる．例えば「かかる」という単語の「前構付品」という素性についての値の集合が判定詞，助動詞，助詞であるとする．同じ単語のある事例の「前構付品」という素性の値が助動詞となっている場合ベクトルは「010」となる．次に各素性値集合についての処理結果を繋げて個々のデータに対応するベクトルを作成する．素性値の集合のサイズはそれぞれ異なるため各単語ごとにベクトルの次元は異なり数千から 1 万 2 千程度となる．

ラベルデータは各事例ごとに語義 ID が与えられており，上記と同様の方法でベクトルに変換する．語義 ID の集合のサイズ，すなわち教師ベクトルの次元数は 2 から 26 までの範囲となっている．

3　Deep Learning

3.1　Denoising Autoencoder

Autoencoder は Deep Learning において教師なし学習で特徴抽出を行うために用いられる．Deep Learning には教師なし学習による特徴抽出に Autoencoder を用いるものと Restricted Boltzmann Machine を用いるものがあるが本研究では Autoencoder を用いることにした．

Autoencoder は図 1 のような 3 層ニューラルネットとして入力と出力を近づけるように学習する．Denoising Autoencoder (dA) では入力ベクトル x を確率的に値を書き変えノイズを付与したベクトル x' によって出力を計算し誤差関数にはノイズ付与前の入力ベクトルを使用する．これによってノイズを取り除くように

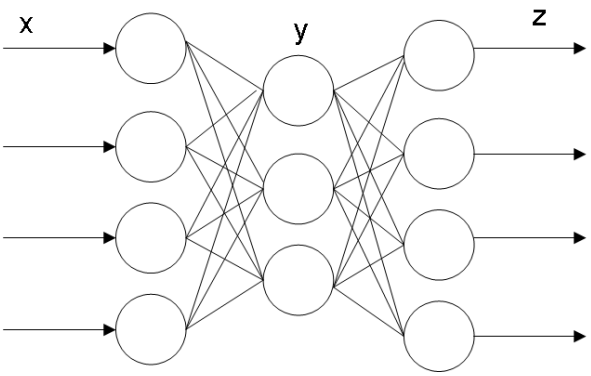

図 1: Autoencoder

学習されて性能が向上する．中間層，出力層の値はそれぞれ (1)，(2) 式によって計算し，交差エントロピー誤差関数による出力層の誤差は (3) 式となる．

$$y_j = \text{sigmoid}(\sum_{i=1}^{n} w_{ji}^{(1)} x_i' + b_j^{(1)}) \tag{1}$$

$$z_j = \text{sigmoid}(\sum_{i=1}^{m} w_{ji}^{(2)} y_i + b_j^{(2)}) \tag{2}$$

$$\text{L}(\text{x}, \text{z}) = \frac{1}{n} \sum_{i=1}^{n} (x \log(z) + (1-x) \log(1-z)) \tag{3}$$

ただし, $w_{ji}^{(1)}$, $b_j^{(1)}$は入力層・中間層間の重みとバイアスであり, $w_{ji}^{(2)}$, $b_j^{(2)}$は中間層・出力層間の重みとバイアスである．

Autoencoder の学習を終えた後中間層と出力層間の重みは使用せず入力層と中間層の 2 層を特徴抽出器として使用する．

3.2 Stacked Denoising Autoencoder

Stacked Denoising Autoencoder (SdA) は Denoising Autoencoder を層ごとに学習し積み重ねて教師ありの学習器をつなげたものである．積み重ねられた Denoising Autoencoder は特徴抽出のために利用され深い層ほどより抽象的な表現を獲得すると考えられている．学習の段階は Pre-trainig と呼ばれる教師無し学習と Fine-tuning と呼ばれる教師あり学習に分けられる．まず入力層に近い層から順に Pre-training を行っていく．ある層を学習させる際に手前の層はパラメータの調整はせずに次層への出力の計算のみを行い，今注目している層のパラメーターのみを調整する．

図 2 の例を挙げて説明すると各層のユニット数が 5-4-5 の dA を学習させそれを dA1 とする．次にユニット数が 4-3-4 の dA を構築し，dA1 の出力を入力として学習させたものを dA2 とする．Pre-training が終了した後は dA2 の出力を MLP や SVM 等の教師あり学習器への入力として学習を行う．

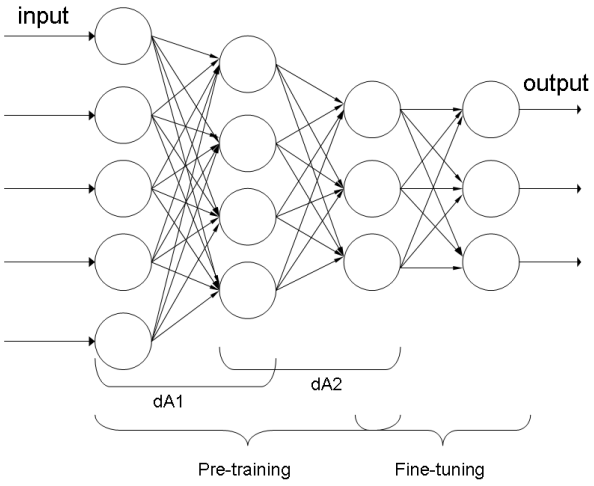

図 2: Stacked Denoising Autoencoder

3.3 正則化

ニューラルネットワークは表現力が高く非常に複雑な関数も表現できる反面，特に深い構造では過学習しやすいという欠点がある．過学習を抑えるために用いられる手法が正則化 [3] である．これを抑えるために重みの絶対値が大きくなり過ぎないように制限をかける weight decay を用いるのが一般的である．weight decay には L1 正則化と L2 正則化があり誤差関数にペナルティ項をあたえることにより重みに制限をかける．バイアス項も同様に正則化できるが，正則化をかけない方がよい場合もある．L1 正則化項は (4) ，L2 正則化項は (5) であり，これらはそれぞれ (3) 式の誤差関数に加えて用いる．

$$\sum_k \sum_j \sum_i \lambda_1 |w_{ji}^{(k)}| \tag{4}$$

$$\text{sqrt}(\sum_k \sum_j \sum_i \lambda_2 (w_{ji}^{(k)})^2) \tag{5}$$

ただし,λ_1は $L1$ 正則化項の係数であり,λ_2は $L2$ 正則化項の係数である。

それぞれの特徴として L1 正則化はスパース性が高く重みを 0 に近づける力が強いため学習後に無駄な計

表 1: 実験結果

学習器	構造	Pre-train_lr	Fine-tune_lr	corruption_level	正解率
SVM					0.778
MLP	a-a/4-b		0.1		0.776
MLP	a-a/16-b		0.1		0.776
SdA(LR)	a-a/4-b	0.01	0.1	[0.1]	0.706
SdA(LR)	a-a/4-a/16-b	0.01	0.1	[0.1,0.2]	0.767
SdA(LR)	a-a/4-a/16-a/64-a/256-b	0.002	0.01	[0.1,0.2,0.3,0.3]	0.772
SdA(MLP)	a-a/4-a/4-a/16-b	0.01	0.1	[0.1,0.2]	0.776
SdA(LR)	a-a/4-a/16-a/64-b	0.002	0.01	[0.3,0.3,0.3]	0.780

算を省き高速化することができる．L2 正則化は精度は高いが過学習しやすい．

4 実験

4.1 実験条件

SdA の教師あり学習にはロジスティック回帰 (Logistic Regression, LR) を用いるものと三層パーセプトロン (MLP) を用いるものの 2 通り試した．MLP 及び SdA での実験ではベクトルの次元が大きく学習に時間がかかることからモデルの評価に cross validation は使用せず，学習の反復回数を 10 回，20 回，30 回，... と 10 回刻みで変化させたものをそれぞれ評価用データで評価し正解率の最高値をそのモデルの精度とする．分類に使用するベクトルの次元は単語ごとに異なり，単語ごとに個別に MLP 及び SdA の各層のユニット数を決定するのは現実的ではない．そのため入力ベクトルの次元数を a，出力ベクトルの次元数を b と表し a-a/2-b のように各単語の学習モデルのユニット数を決定した．また，層の構成や学習率等のハイパーパラメーターの決め方は計算時間の問題によりグリッドサーチやランダムサンプリングは使用せず適当に決定した．

4.2 実験結果及び考察

実験による各学習モデルの学習用データ，評価用データそれぞれの平均正解率を表 1 に示す．教師なし学習での学習と教師あり学習での学習係数を別に設定しそれぞれ Pre-train_lr，Fine-tune_lr としている．SdA には正則化を使用せず，単独の MLP 及び SdA の中の MLP には L2 正則化のみを使用し正則化係数を 0.0001 とした．また SdA においては確率的に入力ベクトルの要素を 0 とする．0 とする確率 (corruption_level) は各層ごとに設定する．MLP と SdA で同じ構成での実験や SdA の層を深くした場合や corruption_level を変化させた場合の実験を行った．

表 1 に示している通り，MLP，SdA，SVM は三者とも同程度の精度となり，先行研究 [4] の最高精度のシンプルベイズや決定リストと比べても精度に大きな差はなかった．MLP，SdA 共に学習用データに対する正解率はほぼ 1 となっているのに対し評価用データに対する正解率が低いことから過学習していると考えられる．

表 2: 再評価による正解率

閾値	0.3	0.4	0.5
SVM	0.789	0.799	0.819
MLP	0.778	0.788	0.807
SdA	0.784	0.794	0.813

原因としてベクトルの次元が数千から 1 万 2 千程度と非常に大きいのに対し各単語の学習用データが 100 個しかないのは十分に学習するには少なすぎることが考えられる．また，学習用データの質が悪く十分に学習させることが不可能な単語が含まれており各学習器の性能比較に悪影響を与えている可能性も考え，MLP，SdA，SVM の評価用データに対する正解率が三者とも閾値以下となる単語を除外し再評価を行った．再評価の結果を表 2 に示す．閾値には 0.3，0.4，0.5 を使

用しそれぞれ再評価を行ったが三者とも正解率に大きな差はなかった．また，入力ベクトルの次元数と正解率に関係があるかを調べるために相関係数を計算したが両者に相関は見られなかった．

5　おわりに

MLP，SdA，SVM を用いて実験し比較を行った．今回実験した結果では三者とも同程度の精度となった．Deep Learning については Dropout 等の様々な手法が提案されておりそれらを取り入れることによってより高精度となる可能性がある．また教師無し学習が行える Deep Learning の特徴を活かした案としてラベル無しデータを増やすことが考えられる．人手によるラベル付けが必要なラベルありデータと異なり，ラベル無しデータは自動的に収集可能であり教師無し学習に利用することができる．この手法は手書き文字認識のデータセット MNIST では MLP 及び SdA には同じデータ数で教師あり学習を行い，SdA の教師なし学習にはより多くのデータを使用する実験において SdA が MLP の精度を上回った．この結果からも上記の手法を本研究にも適用することで精度が向上することが期待できる．

謝辞

本研究は科研費 (25330368) の助成を受けたものである．

参考文献

[1] Yoshua Bengio: Learning Deep Architectures for AI, Foundations and Trends in Machine Learning, Vol.2, No.1, pp. 1-127, 2009
[2] 白井清昭: SENSEVAL-2 日本語辞書タスク，自然言語処理，Vol.10, No.3, pp.　3-24, 2003
[3] 正則化 http://ibisforest.org/index.php?
[4] 村田真樹，内山将夫，内元清貴，馬青，井佐原均： SENSEVAL2J 辞書タスクでの CRL の取り組み日本語単語の多義性解消における種々の機械学習手法と素性の比較，自然言語処理，　Vol.10, No.3, pp. 115-133, 2003