

需要予測におけるニューラルネットワークの活性化関数の 選定に関する研究

李 子揚

LI ZIYANG

キーワード：ロジスティクス、需要予測、活性化関数、精度向上

1. 序論

リーマンショックやコロナショックといった世界経済に急激な影響を与える場合は困難としても、日々の需要の変動に合わせた供給の調整は、過剰在庫や欠品による販売機会の損失を防ぎ、企業の収益を確保する上でとても重要となっている。そのため企業においては、需要予測の精度向上に力をいれており、支援するためのツールが数多く存在している。

このような状況の中、新しい手法としてニューラルネットワークを用いた需要の予測手法が開発され、大きく予測精度を向上させている。しかし、その多くは特定の商品を対象とした需要予測に焦点を当てており、そもそものニューラルネットワークを用いた予測モデルを構築する上で重要な活性化関数の選択については十分な検討がなされていない。

そこで本研究では、活性化関数として、sigmoid、tanh、Relu の 3 つを対象とし、各活性化関数を用いた場合の予測精度と予測時間の比較を行う。そして、この比較結果から活性化関数の選択が、需要予測の精度と予測時間に与える影響について考察することを目的とする。

具体的には、入力層と一つの隠れ層、そして、出力層の三層から構成されるニューラルネットワ

ークを対象に、入力層と隠れ層、隠れ層と出力層、それぞれの間で使用する活性化関数の組み合わせを変えた場合の比較を行い、予測精度と予測時間から望ましい活性化関数の組み合わせについて検討を行う。

2. 本論

2.1 活性化関数

活性化関数は、人工神経ネットワークモデルが非常に複雑で非線形の関数を学習および理解するために非常に重要である。それらは私たちのネットワークに非線形特性を導入した。式(3)と式(4)のように、活性化関数は隠れ層と出力層で使う、

$$Z = g(X * W_1 + b_1) \dots \dots \dots (1)$$

$$\hat{y} = g(Z * W_2 + b_2) \dots \dots \dots (2)$$

Z は $Z_1, Z_2 \dots Z_{d1}$ の組み合わせた行列式、 $\hat{y}$ は y の予測値、 X は $[x_1, x_m]$ の組み合わせた行列式、 W_1 W_2 は重み係数 w_1 w_2 の行列式、 b_1 b_2 はバイアスの行列式、 $g()$ は活性化関数。

実際のデータでは、すべてのパラメーターが線形に関連しているわけではない。これらの非線形

に関連するデータをトレーニングして適合させるには、ニューラルネットワークで非線形の活性化関数を追加する必要がある。活性化関数が追加されていない場合、ニューラルネットワークから出力したのは線形関数。実験により、ニューラルネットワークにいくつの層があっても、最終的な計算は線形関数にすぎず、非線形データに適合できないことが証明されている。

2.2 モデル構築

今回は python でニューラルネットワークを構築することである。商品需要の現状を分析した上で、さまざまな学者の研究成果と個人の理解に基づいて、要因をまとめて分析した結果として、配送方式、オーダーコスト、天気、売上高、購買サイクルなどそれぞれの要因がある。本研究では、ニューラルネットワークに線形と非線形データの損失を比較するため、過去の研究に基づいて、a1、a2、a3、a4 四つの要因数があることを決めた。

ニューラルネットワークの構造について、層数は三層の構造で、出力層と入力層は全部一層で、隠れ層は式(3)のように計算して、四ユニット数を決定した。

$$S = a + \sqrt{m + n} \dots\dots\dots (3)$$

S は隠れ層のユニット数;m は入力層のユニット数;n は出力層のユニット数;a は 1 から 10 までの整数。

活性化関数の選択について、sigmoid 関数、tanh 関数と relu 関数 3 つの他の実験と既存研究でよく選択される活性化関数を選択した。

他のパラメータの設定は以下となる；学習率が異なればフィッティング結果も異なることを考慮し、経験的な判断により、0.1、0.01、0.001 の 3 つの学習率で実験を行い、データが最も良いグループを実験として選択することにした。設定されたサイクル数は 10,000 である。10,000 回の計算後、取得された最終トレーニング損失、トレーニング時間とテスト損失を記録する。

2.3 データ作成

今回のデータは、テクノロジー企業の製品を仮想オブジェクトとして使用して、24 か月のデータを構築して、データは 3 つの異なる内部ロジックを介して確立され、3 つのロジックはそれぞれ合計 9 つのデータを構築した。以下は三つのロジッ

ク：

(1) 変数は、一定の値に基づいてランダムな数値を加算して得られるほぼランダムなデータだ。

(2) 変数は、a2 を除いて、上記 (1) と同じ方法で作成され、a2 は線形的に減少する方法で作成された。

(3) すべての変数は線形に関連する方法で構築されるため、すべてのデータは線形に関連する属性を持っている。

本稿では、データをトレーニングデータとテストデータに分割して、24 か月の元のデータから、6 か月のデータがテストデータとしてランダムに選択され、残りの 18 か月のデータが初期トレーニングデータとして使用された。

次に、データの相関係数を計算して、データの相関関係を示している。

図 1 相関係数

a1	a2	a3	a4
0.686951	0.100618138	0.376775623	0.376775623
0.737394	-0.00387795	-0.42116635	-0.12544292
0.091229	-0.13959629	-0.12080737	-0.250916757
0.6615	0.167057363	-0.01986571	0.192967411
0.55103	-0.14888496	0.118845198	-0.405109544
0.624255	-0.18241951	-0.11895095	-0.079098472
0.516048	0.074681218	0.168297678	-0.181672659
0.563581	-0.23997994	0.104686073	0.059760254
0.319797	0.107444296	-0.03493125	0.101767921

図 1 は九つのデータで、四つの独立変数と販売量の相関係数である。

データを正規化したから計算することが可能となっている。今回使用されている標準化は標準得点であり、具体的な計算方法は：

$$z = \frac{x - \mu}{\sigma} \dots\dots\dots (4)$$

z は標準得点、x は変数の原始値、μ は変数の平均値、σ は変数の標準偏差。

2.4 実験結果

図 2、図 3 と図 4 はデータをニューラルネットワークに投入して訓練した結果となっている。

図 2 トレーニング損失

sig*sig	sig*tanh	sig*relu	tanh*sig	tanh*tanh	tanh*relu	relu*sig	relu*tanh	relu*relu
0.483	0.058	0.419	0.277	0.14	0.326	0.498	0.155	0.376
0.499	0.06	0.418	0.322	0.057	0.423	0.499	0.379	0.499
0.483	0.193	0.424	0.326	0.143	0.339	0.491	0.293	0.417
0.486	0.109	0.39	0.372	0.162	0.338	0.484	0.293	0.477
0.49	0.038	0.36	0.317	0.022	0.323	0.499	0.443	0.499
0.489	0.122	0.356	0.324	0.109	0.326	0.497	0.393	0.499
0.489	0.098	0.343	0.316	0.078	0.32	0.502	0.092	0.391
0.447	0.074	0.293	0.283	0.077	0.291	0.49	0.068	0.326
0.451	0.102	0.305	0.281	0.065	0.254	0.469	0.107	0.294

図 3 訓練時間

sig*sig	sig*tanh	sig*relu	tanh*sig	tanh*tanh	tanh*relu	relu*sig	relu*tanh	relu*relu
28423	30255	24285	24936	23384	24816	23917	24226	26914
27890	30316	24318	25018	23789	24667	23994	24299	26357
27531	30127	24056	24841	23254	24549	23794	24245	25345
27813	29673	23566	25057	23287	24732	23803	24493	25778
26862	30920	23920	25271	24011	24585	23654	24416	24990
27745	30141	23891	25155	23784	24543	24615	24634	24887
27719	29380	24195	25046	24147	24327	23610	24504	24966
27319	30054	24566	25407	24062	24709	23574	24286	24871
27781	30003	24458	24856	24660	24547	23865	24114	24946

図 4 テスト損失

sig*sig	sig*tanh	sig*relu	tanh*sig	tanh*tanh	tanh*relu	relu*sig	relu*tanh	relu*relu
0.733	0.109	0.676	0.352	0.313	0.313	0.741	0.274	0.525
0.397	0.01	0.397	0.286	0.305	0.305	0.397	0.181	0.397
0.609	0.221	0.546	0.286	0.339	0.339	0.649	0.331	0.561
0.49	0.145	0.32	0.188	0.196	0.196	0.651	0.322	0.479
0.526	0.05	0.413	0.381	0.368	0.368	0.544	0.546	0.537
0.447	0.036	0.447	0.447	0.447	0.447	0.451	0.226	0.447
0.876	0.103	0.834	0.834	0.834	0.834	0.834	0.103	0.834
0.752	0.128	0.752	0.752	0.752	0.752	0.76	0.132	0.752
0.599	0.014	0.599	0.599	0.599	0.599	0.599	0.021	0.599

トレーニング損失について、sigmoid の他の組み合わせも比較的悪い結果で、tanh と組み合わせただけの場合にのみ、比較的良い結果が得られ、結果は安定する傾向がある。したがって、通常 sigmoid 関数を活性化関数として使用することはお勧めできず、安定性が必要な場合にのみ使用する。tanh 機能の全体的なパフォーマンスは、特に tanh×tanh の組み合わせの方が優れている。また、tanh の結果範囲は $[-1, 1]$ で、tanh の後に relu 関数を使用すると、ニューラルネットワークの密度が緩くなり、計算の精度が向上できる。ただし、relu 関数の不確実性が高いため、より良いモデルを取得するには、適切な学習率を選択し、テスト計算を何度も繰り返す必要がある場合がある。

隠れ層と出力層の活性化関数を選択する優先順位について、平均損失表から、出力層の活性化関

数として tanh を使用すると、relu 自体はランダムデータに敏感ではないため、relu との組み合わせを除いて、フィッティング効果が最も高くなることがわかった。機能の他の組み合わせについては、隠れ層と出力層の選択に大きな違いがない。

時間について、表から、sigmoid 関数の組み合わせには時間がかかりかかることがわかった。relu×relu 関数の組み合わせを除いて、relu との他の組み合わせで費やされる時間は比較的短いで、relu 関数の不安定性を考慮して、複数回の試行に十分な時間があれば、relu 関数を使用することが可能です。

テストデータについて、ほぼトレーニングデータと一致する結果を得ることができる。つまり、出力層で tanh を使用すると、最適な結果を得ることができる。トレーニングデータとの違いは、出力層で tanh を使用することの組み合わせを除いて、線形に関連するデータでは適切な結果が得られないことだ。これは、線形に相関するデータからランダムにデータを抽出したことによって、全体的な損失が高くなる。

データタイプに関しては、training 損失で大きな組み合わせに対して、線形データのフィッティング効果がいい。ただ、テスト部分では、線形データはランダムデータよりフィッティング効果は非常に劣る。

最後の結論として、種々のデータを用いて、各活性化関数の組み合わせについて検討を行った結果、sigmoid 関数を用いた場合に予測精度が安定していることが分かった。しかし、他の活性化関数に比べてその予測精度は良くないため、精度を重視する場合の使用は望ましくないことが分かった。次に、tanh 関数は優秀で全体的に予測精度も良く、安定していることが分かった。最後に、Relu 関数は予測精度が安定せず、良い精度とするためには適切な学習率の選択が重要となることが分かった。特に、Relu×Relu の組み合わせの場合は注意が必要で、予測精度が安定しないため組み合わせとして使用することは望ましくないことが分かった。

3. まとめ

本稿では、python を使用してニューラルネットワークを構築し、隠れ層と出力層でそれぞれ異なる活性化関数を選択し、活性化関数の組み合わせ

によって、どの組み合わせが最小の損失を生成し、最短の時間にかかるかを判断する。最後に、各活性化関数に対して、利点と欠点を分析し、使用すべきな場所と時間に対する意見を述べた。

謝辞

本論文を作成するにあたり、ご指導・ご協力をいただいた方々に感謝の意を表します。

黒川先生には、研究の進め方や枠組みについて有益な助言をいただきました。黒川研究室の皆様には、温かいご指導ご鞭撻を賜りました。心より感謝申し上げます。

参考文献

- (1) 小野田 崇・大場 英二：翌日最大電力需要予測における最適なニューラルネットワーク構成の決定法，電気学会論文誌，Vol. 118，No. 5，p. 497-504，1998. 5.
- (2) 麻生 英樹：ニューラルネットワーク情報処理，産業図書，1998.
- (3) 千葉 周一：食品小売デリカ部門の需要予測と発注管理による食品ロス対策，電気設備学会誌，Vol. 40，No. 10，p. 638-641，2020.
- (4) 田辺 正志・吉田 律：ニューラルネットワークによる在庫予測の精度，経営学紀要，Vol. 1，No. 2，p. 127-133，1994. 1.
- (5) 麻生英樹：ニューラルネットワーク情報処理，産業図書，1998.
- (6) 赵馨宇・黄福珍・周晨旭：基于 ReLU 稀疏的卷积神经网络的 MAXOUT 卷积神经网络的数据分类算法，上海電力大学学报，· Vol. 36，No. 3，2020. 6.
- (7) 刘楚辉：基于 BP 网络的连锁超市供应链库存预测和控制研究，甘肃政法学院，2019. 5.
- (8) 腾杨刚・陈劲杰・葛桂林：基于 PCA 主成分分析和 BP 神经网络企业库存预测的研究，软件工程 SOFTWARE ENGINEERING，Vol. 21，No. 7，2018. 7.
- (9) David Simchi-Levi : Designing & Managing the Supply Chain, p305-307, 2010.
- (10) Baiquan Lu・Junichi Murata・Kotaro Hirasawa: A New Method Based on Determining Error Surface for Designing Three Layer Neural Networks, 計測自動制御学会論文集, Vol.36, No.7, 589-598, 2000.
- (11) Cousera, Neural Networks and Deep Learning, <https://www.coursera.org/learn/neural-networks-deep-learning/home>, 2020.
- (12) Edx, Supply Chain Mnaagement, <https://www.edx.org/micromasters/mitx-supply-chain-management>